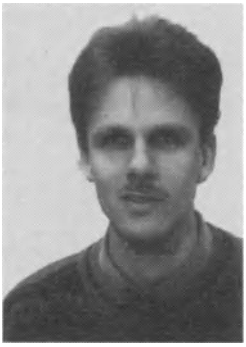
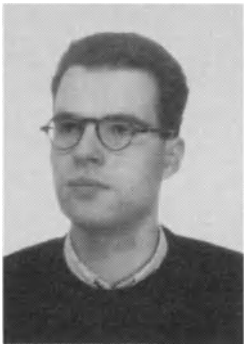
LINUX – vom PC zur Workstation





Die Autoren Stefan Strobel und Thomas Uhl studieren Medizinische Informatik an der Universität Heidelberg/ Fachhochschule Heilbronn.
Sie betreuen seit längerem die an der Hochschule installierten Workstations und Linux-PCs.
In mehreren Vorträgen machten sie Linux einer breiten Öffentlichkeit bekannt und unterstützten dessen Verbreitung.

Stefan Strobel Thomas Uhl

LINUX

Vom PC zur Workstation

Grundlagen, Installation und praktischer Einsatz

Mit einem Vorwort von Jürgen Gulbins
Mit 51 Abbildungen

Springer-Verlag
Berlin Heidelberg New York
London Paris Tokyo
Hong Kong Barcelona
Budapest

Stefan Strobel
Schlegelstraße 19
D-74074 Heilbronn

Thomas Uhl
Obere Heerbergstraße 17
D-97078 Würzburg

Die Autoren sind unter der folgenden E-Mail-Adresse zu erreichen:
linux@sun1. rz.fh-heilbronn. de

ISBN 978-3-540-57383-8 ISBN 978-3-642-97535-6 (eBook)
DOI 10.1007/978-3-642-97535-6

Die Deutsche Bibliothek – CIP-Einheitsaufnahme
Strobel, Stefan: LINUX – vom PC zur workstation: Grundlagen, Installation und praktischer Einsatz / S. Strobel; T. Uhl. Mit einem Vorw. von Jürgen Gulbins. – Berlin; Heidelberg; New York; London; Paris; Tokyo; Hong Kong; Barcelona; Budapest: Springer, 1994

Umschlaggestaltung: Konzept & Design, Ilvesheim
Satz: Reproduktionsfertige Vorlage vom Autor
SPIN: 10131439 33/3140 – 5 4 3 2 1 0 – Gedruckt auf säurefreiem Papier

Vorwort

UNIX hat seine starke Verbreitung, seine Durchdringung des Hochschulbereichs und von dort aus auch der Forschungsanlagen und der Industrie durch den Umstand erlangt, daß es von AT&T anfänglich relativ frei und in Quellen an alle Interessenten weitergegeben wurde. Die heutige UNIX-Funktionalität wurde so nicht nur von den AT&T-Entwicklern geschaffen, sondern ebenso von anderen Entwicklern außerhalb, die das Produkt benutzten und weiterentwickelten – ihre Erweiterungen aber in die AT&T-Entwicklung zurückfließen ließen. Als Beispiel seien hier nur die Beiträge der Universität von Berkeley (Kalifornien) angeführt. Mit der seit etwa 1983 stattfindenden Kommerzialisierung von UNIX durch AT&T (inzwischen USL) wurde diese kreative und kooperative Weiterentwicklung leider zunehmend eingeschränkt, und die USL-UNIX-Quellen sind heute unerschwinglich teuer und praktisch kaum noch zugänglich.

Mit Linux wird den interessierten Informatikern und Computer-Anwendern – und ebenso auch den -Innen – erneut ein System in die Hand gegeben, das diese alte UNIX-Tradition erneut aufleben läßt: Linux ist frei verfügbar und alle sind herzlich aufgefordert (aber nicht verpflichtet) zu seiner Weiterentwicklung beizutragen. Es beginnt, da es auf PC-Systemen läuft, die Arbeitszimmer vieler Studenten und Informatik-Interessierten zu erobern.

Die inzwischen erreichte Funktionalität und Vollständigkeit sind dabei weit jenseits eines reinen Spiel- oder Demonstrationssystems. Linux kann sich heute in vielen, wenn auch vielleicht nicht in allen Aspekten mit den kommerziellen und realistisch gesehen proprietären UNIX-Versionen von SUN, HP, IBM, SCO und anderen durchaus messen. Die erfolgreichen Portierungen ei-

ner ganzen Anzahl kleiner und mittlerer Applikationen und die Verfügbarkeit der aktuellen X11-Versionen demonstrieren dies anschaulich.
Free Software wie Linux, hinter der keine durch Beiträge, Subventionen oder kommerzielle Verkäufe abgestützte Finanzierung steht, hat jedoch mit einigen Problemen zu kämpfen. So fehlt bei ihr z.B. oft eine ausgefeilte, vollständige Dokumentation. An diesem Punkt möchte dieses Buch bei Linux ansetzen. Es will die durchaus vorhandenen Linux-Manual-Pages durch eine Übersicht und durch zahlreiche Hinweise und Vervollständigungen ergänzen. Es gestattet der Linux-Anwenderin und dem Linux-Anwender sich damit schneller in Linux einzuarbeiten, das System besser zu nutzen und auch einmal in Bereiche hineinzuschauen, die sie oder er bisher noch nicht kennt oder gefunden hat.
Insbesondere die Hinweise zur Installation und Administration – beides ist bisher in UNIX leider von System zu System sehr verschieden – dürften den meisten Linux-Benutzern hilfreich sein.
Ich würde mir wünschen, daß mit Linux wieder der Pioniergeist der früheren UNIX-Jahre auferstehen und den in vielen Aspekten unsinnigen, offenen und in Wirklichkeit proprietären Diskussionen kommerzieller UNIX-Macher aufzeigt, was möglich und sinnvoll ist. Mag dieses Buch dazu beitragen.

J. Gulbins

Anmerkung der Autoren

Im Gegensatz zu vielen anderen Autoren haben wir nicht versucht, die englischen Fachbegriffe ins Deutsche zu übersetzen, da dies nicht zu einer besseren Verständlichkeit der Thematik beiträgt, sondern eher zu Mißverständnissen führt. Beispiele hierfür sind Filesystem oder Manual page. Dies führt unter Umständen dazu, daß in den ersten Kapiteln Begriffe benutzt werden, noch bevor sie näher erläutert werden.

Danksagungen

Die Autoren möchten sich ausdrücklich bei folgenden Personen bedanken, die tatkräftig zur Entstehung dieses Buches beigetragen haben: Dirk Höfle, Sascha Runge, Peter Welker und Roland Uhl. Besonderer Dank für die ausgezeichnete Unterstützung gebührt dem Rechenzentrumsleiter der Fachhochschule Heilbronn, Dr. G. Peter, und seinen Mitarbeitern. Herrn Gulbins danken wir für die Bereitschaft, das Vorwort zu verfassen.

Inhalt

Einleitung

Grundlagen

Vernetzung

Linux Features

Installation

Konfiguration

Administration

Support & Hilfe

X-Windows

Sprachen & Tools

Anwendungen

Netzanwendungen

Anhang

Einleitung

Seit einiger Zeit sind 32-Bit Betriebssysteme eines der meist diskutierten Themen in der Welt der PCs. Es sieht fast so aus, als werde MS-DOS in absehbarer Zeit von leistungsstärkeren Betriebssystemen abgelöst.

Die Diskussion über den zukünftigen Standard ist inzwischen, zumindest in den Fachzeitschriften, lebhaft im Gange. Dabei scheinen sich im Bereich der Server-Betriebssysteme zwei Alternativen abzuzeichnen: Window NT und UNIX, in Varianten wie Solaris 2.1, UnixWare, NextStep 486 und anderen. OS/2 spielt in diesem Zusammenhang keine große Rolle, da es eher als Konkurrenz zu Windows in der jetzigen und den künftigen 32-Bit Versionen zu sehen ist.

UNIX vs Windows NT

Welches System sich letztlich durchsetzen wird ist heute noch nicht absehbar. Jedenfalls ist aufgrund der in den letzten Jahren erheblich gesteigerten Leistung im Bereich Hardware der Bedarf für ein modernes Betriebssystem vorhanden, das dieser Entwicklung Rechnung trägt.

Damit werden die Grenzen zwischen klassischen UNIX- Workstations und High End PCs unter einem modernen Serverbetriebssystem fließend.

1.1 Geschichtliches zu Linux

Abseits aller großen Strategiedebatten hat sich eine enorm leistungsstarke Alternative zu den oben genannten kommerziellen Systemen entwickelt. Die Rede ist von Linux, einem komplett kostenlos erhältlichen UNIX-System für Intel-Prozessoren.

Linux wurde von einem jungen finnischen Studenten namens Linus Torvalds entwickelt. Es war jedoch zunächst wohl nicht sein Ziel, ein vollwertiges Betriebssystem zu implementieren.

386-Prozessor

Anfangs wollte er nur die speziellen Task-Switch-Befehle des 80386-Prozessors näher kennen– und verstehen lernen. Er benutzte zunächst MINIX, ein Lehrsystem für Betriebssystembauer von Andrew Tannenbaum, um sein Testprogramm zu übersetzen. Doch MINIX enthielt, aufgrund seiner eher didaktischen Ausrichtung, eine Menge Unzulänglichkeiten.

Dem ehrgeizigen Studenten reichten bald die vorhandenen Möglichkeiten dieses UNIX-ähnlichen Systems nicht mehr aus. Er begann aus seinen Testprogrammen Schritt für Schritt einen kleinen Betriebssystemkern (Kernel) zu entwickeln, der im sogenannten Protected Mode des 80386ers lief und diesen Prozessor somit optimal ausnutzte.

Nach dem Task-Switcher folgte ein einfacher Tastatur-Treiber, um interaktiv mit dem System arbeiten zu können. Zu diesem Zeitpunkt war Linux noch immer auf Teile des MINIX-Systems angewiesen, was sich aber rasch ändern sollte.

Um nicht auch noch ein neues Dateisystem entwickeln zu müssen, entschloß sich Linus Torvalds, das Filesystem von MINIX zu übernehmen. Auf diese Weise ersparte er sich nicht nur eine Menge Arbeit, sondern es stand ihm von Anfang an ein stabiles System zur Festplattenverwaltung zur Verfügung. Nach wenigen Monaten hielt der Autor das System für ausgreift genug, um es einer breiteren Öffentlichkeit vorzustellen.

Im August 1991 erschienen die kompletten Quelltexte von Linux Version 0.01 erstmalig auf dem größten finnischen FTP-Server im Internet nic.funet.fi. Es wurde als "freely distributable Minix clone" angekündigt und nur von wenigen Interessierten im Internet beachtet. Bereits im Oktober wurde die nächste Version (0.02), die auch schon einige rudimentäre UNIX-Kommandos beinhaltete, veröffentlicht. Der dazu verfügbare GNU C-Compiler (gcc) erlaubte die Übersetzung kleinerer C-Programme und ermöglichte so die Portierung einer UNIX-Shell (bash).

POSIX

Eine entscheidende Rolle für die einfache Übertragbarkeit von Standard UNIX-Software auf das heutige Linux spielte die frühe Entscheidung, sich an POSIX zu orientieren; einer Familie von

Standards des *Institute of Electrical and Electronics Engineers* (IEEE). Es dauerte aber noch bis zum Ende des Jahres, ehe Linux größere Beachtung fand. Der Durchbruch kam am 5. Januar 1992 mit der Version 0.12. Linux war nun mächtig genug, um eine größere Entwickler-Gemeinde zu interessieren. Inzwischen verfügte das System schon über einen Swapping-Mechanismus, der es gegenüber MINIX eindeutig überlegen machte.
Im Laufe der Zeit fanden sich immer mehr interessierte Entwickler, die Fehlerkorrekturen und Verbesserungsvorschläge nach Finnland schickten und auf diese Weise mit an der Verbesserung des Systems arbeiteten. Solche frühen Fremdentwicklungen sind beispielsweise die in dieser Version enthaltene "POSIX Job Control" oder die umschaltbaren virtuellen Konsolen.
"Linux" wird sehr oft falsch ausgesprochen. Viele Anwender halten den Namen für einen amerikanischen Begriff und reden daher von "Lainux". Richtig ist jedoch die finnische Aussprache. Diese entspricht etwa dem, was ein Deutscher ohne Englischkenntnisse sagen würde, also "Lihnucks".

Aussprache

Als wichtiges Instrument für die rasche Fortentwicklung stellte sich das Internet heraus. Dabei handelt es sich um ein weltweites Netzwerk (WAN), in dem über eine Million Computer miteinander verbunden sind. Es erlaubt den schnellen Austausch von Informationen aller Art. Über dieses Netzwerk konnten die Linux- Entwickler ihre Kommentare, Verbesserungen und Programme austauschen.

Internet

Zu Anfang wurde Linus Torvalds täglich mit mehr als 60 Mails überhäuft, die er kaum noch alle bearbeiten und beantworten konnte. Erst als eine eigene Diskussionsgruppe für Linux eingerichtet wurde, beruhigte sich die Flut an Nachrichten wieder. Heute existieren im Internet mehrere verschiedene News-Groups, die sich mit Linux befassen. Die wichtigste ist `comp.os.linux.announce` (c.o.l.a.), in der neue Entwicklungen und Programmversionen angekündigt werden.
Für die Entwickler wurden sogenannte Mailing-Lists eingerichtet, die einen ähnlichen Informationsaustausch zulassen. Um eine bessere Auswahl nach speziellen Gebieten zu ermöglichen,

wurden verschiedene Kanäle eingerichtet, in denen jeweils nur bestimmte Themen diskutiert werden (Kernel, X11, Netzwerk, Compiler, usw.).

Neben Briefen und Nachrichten können im Internet auch Dateien ausgetauscht werden, so daß sich auch die verteilte Entwicklung größerer Software-Pakete organisieren läßt, wie das Beispiel Linux eindrucksvoll demonstriert.

Support

Aber nicht nur für die Entwickler ist der schnelle Informationsfluß von Vorteil, auch der Anwender hat greifbare Vorteile. Sollten bei der Installation oder beim Betrieb irgendwelche Probleme auftauchen oder Fehler entdeckt werden, so kann er in günstigen Fällen innerhalb weniger Stunden eine adäquate Lösung für sein Problem erhalten. Eine derartige Unterstützung gewährleisten oft nicht einmal kommerzielle Wartungsverträge.

Mailboxen

Natürlich hat nicht jeder Linux-Anwender einen Internet-Zugang. Doch auch in diesen Fällen ist dieser nicht auf sich alleine gestellt. In den meisten Mailbox-Netzen wurden inzwischen Diskussionsgruppen über Linux eingerichtet, so daß in vielen Fällen ein Modem genügen sollte, um einigermaßen auf dem Laufenden zu bleiben.

Das Interessante an der Geschichte von Linux ist, daß keine strenge Hierarchie und eine übergeordnete Instanz existiert, die die Entwicklung in irgendeiner Weise steuert. Vielmehr wird das Projekt vom Enthusiasmus vieler einzelner Internet-Teilnehmer getragen, die immer neue Verbesserungen und Vorschläge einbringen. Es sind häufig professionelle Entwickler oder Angestellte größerer Institutionen, die dafür ihre Freizeit opfern.

Zwar liegt die konkrete Weiterentwicklung des Kernels noch immer in den Händen des ursprünglichen Autors, aber es haben sich inzwischen einige weitere kompetente Mitstreiter gefunden, die sich um andere Teilbereiche des Systems kümmern.

GNU C und X-Windows

Ein solcher Bereich ist beispielsweise die Portierung bzw. Wartung des GNU C-Compilers und der C-Libraries für Linux, die Pflege und Anpassung des X-Window-Systems oder des Netzwerkbereiches. Wieder andere Linux-Anhänger arbeiten an einer Benutzer- und Systemdokumentation oder sorgen für die Zusammenstellung eines installationsfähigen Systems auf Disketten oder CD-ROM.

Die freie Verfügbarkeit der Quelltexte gilt nicht nur für den Kernel, sondern auch für die meisten Anwenderprogramme. Sie entstammen überwiegend den großen UNIX Freeware–Archiven des Internets.
In größeren Abständen wird eine Softwareliste im Internet verbreitet, die momentan ca. 800 Pakete enthält. Es gibt eigentlich keinen Bereich, für den nicht eine passende Software zu finden wäre. Da vor allem an den amerikanischen Universitäten sehr viele Entwicklungen unter UNIX durchgeführt werden, die dann frei erhältlich sind, gibt es für Linux auch viele Implementierungen aus dem Bereich der Forschung.
Als Beispiel wären diverse Compiler für bekannte und weniger bekannte Programmiersprachen zu nennen. Auch die Datenbanksysteme Ingres und Postgres der Berkeley University wurden inzwischen für Linux portiert.
Noch liegt der Schwerpunkt im Bereich der Freeware, aber es sind auch schon kommerzielle Applikationen verfügbar. So zum Beispiel ein Modula-2 Compiler, ein Smalltalk-Entwicklungssystem, der grafische Interfacebuilder von Parcplace oder die in der UNIX-Welt zum Standard gewordene grafische Benutzeroberfläche OSF/Motif.

OSF/Motif

Die Weiterentwicklung von Linux geht momentan in ähnlich großen Sprüngen voran wie die ersten Implementierungen des Kernels. Version 1.0 sollte im Dezember '92 erscheinen, wurde jedoch auf einen späteren Termin verschoben. Der Grund hierfür war nicht die mangelnde Stabilität, sondern der den kommerziellen UNIX-Versionen noch nicht ebenbürtige Funktionsumfang.

Version 1.0

So wurde das Erscheinen der Version 1.0 immer weiter hinausgezögert. Im nachhinein betrachtet, hätte Linus Torwalds, nach eigenen Aussagen, am liebsten die erste stabile und anwendbare Version 0.12 als Version 1.0 deklariert, denn die Null in der Versionsnummer scheint einige potentielle Interessenten davon abgehalten zu haben, sich näher mit dem System zu befassen.
Die für Anfang '94 erwartete endgültige Version 1.0 wird jedoch mit allen wichtigen Features der kommerziellen Konkurrenten

aufwarten können. Einen Vorteil gegenüber diesen Systemen kann Linux schon heute für sich geltend machen. Aufgrund des effizienten Designs bietet es bei gleicher Hardwarekonfiguration eine weit höhere Performance. Dies gilt nicht nur für den Kernelbereich, sondern auch für die grafische Oberfläche.

X-Windows

Im Gegensatz zu vielen anderen UNIX-Systemen verfügt Linux bereits jetzt die neueste Version des X-Window-Systems X11R5. Zu den Features, die in kommerziellen Systemen nicht ohne weiteres zur Verfügung stehen, zählt z.B. die Unterstützung von INMOS-Transputerboards oder die Möglichkeit TCP/IP über die serielle oder parallel Schnittstelle zu betreiben. Für die Zukunft ist auch die direkte Kernelunterstützung von ISDN-Karten zur schnellen Netzwerkverbindung über größere Distanzen geplant. Da es jedoch keine echte "Roadmap" für die Weiterentwicklung von Linux gibt, wird das System in Zukunft sicher noch mit einigen Überraschungen aufwarten können.

1.2 UNIX-Entwicklung und Standards

Die Geschichte von UNIX reicht weit in die 70er Jahre zurück. Die erste Version wurde 1971 von Dennis Ritchie und Ken Tompson in den Bell Laboratories, der größten amerikanischen Telefongesellschaft AT&T entwickelt.

Als 1973 die aus BCPL und B hervorgegangene Compilersprache C zur Verfügung stand, wurde der größte Teil des Systems neu geschrieben, was sich später als sehr günstig für Übertragungen auf andere Prozessoren herausstellen sollte.

AT&T

Aufgrund einer Vereinbarung mit der US-Regierung durfte AT&T das inzwischen recht erfolgreiche System nicht kommerziell verwerten. Daher wurde es im Quelltext, jedoch ohne Support an Universitäten weitergegeben, wo es zunehmend an Popularität gewann. Mit der Version 7 im Jahre 1979 änderte sich die Lizenzpolitik von AT&T. Den Quelltext konnte man nur noch gegen Gebühren erhalten, was die University of Berkeley später veranlaßte, eine eigene UNIX-Variante, das BSD-UNIX zu entwickeln.

1983 kündigte AT&T eine Weiterentwicklung ihres Systems mit dem Namen System V an, welches nun kommerziell vertrieben

wurde. Die Programmierschnittstelle dieses Systems wurde in der sogenannten *System V Interface Definition* festgelegt.

Firmen wie Sun Microsystems, Microsoft oder DEC entwickelten jeweils ihre eigenen Versionen von UNIX (SunOS, Xenix, ULTRIX), was im Laufe der Zeit die Portierung von Software zwischen diesen Systemen unnötig erschwerte. Um die zwei großen UNIX-Linien wieder zusammenzuführen, propagierte AT&T im Jahre 1990 Release 4 des System V als neuen Standard, der alle bisherigen UNIX-Varianten umfaßte.

System V

Auch andere Institutionen erkannten die Notwendigkeit einer Standardisierung von UNIX. So entwarf das Institute of Electrical and Electronics Engineers (IEEE) den sogenannten POSIX-Standard für UNIX-ähnliche Betriebssysteme. Dieser Standard gliedert sich in mehrere Teile.

So enthält POSIX 1003.1 nur eine Beschreibung der unteren Systemschnittstellen. POSIX 1003.2 wird ein Standard für Shells und Kommandos definieren und POSIX 1003.7 die Möglichkeiten der Systemadministration. Obwohl POSIX eigentlich auf der UNIX-Systemschnittstelle basiert, wird dieser Standard auch von anderen Betriebssystemen unterstützt werden (Windows NT).

POSIX

Ein anderes Gremium, das sich vorwiegend aus UNIX-Herstellern zusammensetzt, hat einen weiteren Standard verabschiedet. Der sogenannte *X/Open Portability Guide* basiert zwar auf POSIX 1003.1, erweitert diesen jedoch in einigen Punkten. Im Rahmen der COSE-Initiative nahm die Bedeutung dieser Organisation erheblich zu. Ziel ist nun die Verabschiedung einer einheitlichen Desktop-Oberfläche und Programmierschnittstelle für alle vorhandenen UNIX-Varianten. Dadurch soll vor allem der Portierungsaufwand von Software erheblich reduziert werden.

1.3 Die Free Software Foundation

Neben der Orientierung am POSIX-Standard zeichnet sich Linux dadurch aus, daß viele Systemkomponenten dem GNU-Projekt der Free Software Foundation (FSF) entstammen und diese Organisation somit einen wichtigen Beitrag zur Entwicklung des Systems geleistet hat.

Die FSF wurde von Richard Stallman, dem Autor des legendären GNU Emacs Editors, gegründet. Ziel dieser Organisation ist es, die üblichen Restriktionen beim Kopieren und Modifizieren von Software abzubauen und die Verbreitung von Free Software zu fördern.

Copyleft

Die genauen Bedingungen werden im sogenannten Copyleft der FSF festgelegt. Software, die dem Copyleft der FSF unterliegt, darf durchaus kommerziell vertrieben werden, muß jedoch komplett im Quelltext weitergegeben werden. Dies gilt nicht für Programme, die zwar mit dem GNU-Compiler entwickelt wurden, jedoch keine der Routinen der GNU-Bibliotheken benutzen.

Das größte Projekt der FSF ist die Implementierung eines frei kopierbaren Betriebssystems, das weitgehend UNIX-kompatibel sein wird. GNU steht übrigens für "Gnu's not UNIX". Im Rahmen dieses Projektes wurde auch der inzwischen auf den meisten Hardwareplattformen erhältliche GNU C- und C++-Compiler entwickelt.

GNU C / C++

Zahlreiche Programmierer in aller Welt haben sich der Idee der Free Software angeschlossen und fördern unter anderem durch die Entwicklung UNIX-kompatibler Kommandos das GNU-Projekt. Mittlerweile findet der interessierte Programmierer auf den GNU FTP-Servern eine beinahe unüberschaubare Fülle an Software. Es stehen, neben zahlreichen UNIX-Kommandos und Programmiersprachen wie C, C++, Smalltalk, Lisp, oder Fortran, verschiedene Editoren, Debugger (gdb) und sogar ein Postscript Interpreter (Ghostscript) zur Verfügung.

Linux stellt zwar keine direkte Entwicklung der FSF dar, dennoch ist diese von entscheidender Bedeutung. Linux unterliegt ebenfalls dem Copyleft und enthält zahlreiche Kommandos, Editoren und Compiler die eigentlich dem GNU-Projekt zugeordnet sind.

1.4 Linux Features im Überblick

Zur besseren Orientierung werden im folgenden die wichtigsten Merkmale von Linux in Stichpunkten zusammengefaßt.

- **Echtes 32-Bit Multiuser/Multitasking UNIX-System.** Linux erlaubt mehreren Benutzern die gleichzeitige Ausführung verschiedener Programme und nutzt dabei die Möglichkeiten der Intel 386 Prozessoren und deren Nachfolger voll aus. Die Performance ist dabei durchaus mit einer klassischen RISC-Workstation vergleichbar.
- **Orientierung an gängigen UNIX-Standards (POSIX).** Durch die Einhaltung der bestehenden Standards für UNIX ist die Portierung von vorhandener Software auf Linux meist ohne Probleme durchführbar.
- **Netzwerkunterstützung (TCP/IP).** Ein Linux Rechner kann auf einfache Weise in ein TCP/IP-Netz integriert werden. Es werden gängige PC-Ethernetkarten und eine TCP/IP-Verbindung über Modems (SLIP) unterstützt.
- **Grafische Oberfläche (X-Windows).** Die aktuelle Version (Release 5) des X-Window-Systems ist im Linux-System enthalten. Mit OSF/Motif steht auch die Standard-Oberfläche der kommerziellen UNIX-Systeme zur Verfügung.
- **GNU-Utilities und Programme.** Ein großer Teil der Befehle und Utilities unter Linux entstammen dem GNU-Projekt und zeichnet sich durch viele funktionelle Erweiterungen aus.
- **Komplette UNIX-Entwicklungsumgebung.** Linux erlaubt die Entwicklung von Programmen, die auch problemlos auf anderen UNIX-Systemen lauffähig sind. Neben den GNU C / C++ / Objective-C Compilern, vielen Editoren und einem Versionskontrollsystem gibt es zahlreiche weitere Tools zur Softwareentwicklung.

Grundlagen

Für das Verständnis der folgenden Kapitel sind einige Grundkenntnisse der EDV im allgemeinen und UNIX im besonderen erforderlich. Um dem mit UNIX unerfahrenen Leser den Einstieg in diese Materie zu erleichtern, sollen hier einige der wichtigsten Konzepte und Begriffe erläutert werden.

2.1 Multiuser

In der klassischen Datenverarbeitung gibt es einen zentralen Großrechner, der alle anfallenden EDV-Aufgaben zu verarbeiten hat. An diesem Großrechner sind über serielle Verbindungen Terminals (einfache Textbildschirme mit einer Tastatur) angeschlossen.

Zentralrechner

Da sich viele Benutzer den selben Rechner teilen müssen, ist es nötig, ein System für den Zugangsschutz und die Verwaltung dieser Benutzer zu entwickeln, um eine gerechte Verteilung der gemeinsam genutzten Ressourcen zu erreichen.

Die Basis für ein solches System bilden eindeutige Benutzernamen, die auch *User Ids* genannt werden. Jeder Benutzer bekommt eine solche User Id zugewiesen, mit der er sich beim System mit einem zusätzlichen Passwort anmelden muß. Man spricht hier auch von "Einloggen" in ein System. Aufgrund dieser Kennung können dann auch Zugriffsrechte für Dateien und andere Ressourcen vergeben werden.

Benutzer, User Id, Passworte

Systeme, die eine derartige Verwaltung für mehrere Benutzer bieten, werden Multiuser-Systeme genannt und UNIX ist ein typisches System dieser Art. Es sieht für jeden Benutzer eine

Benutzerkennung (User Id) und mehrere Gruppen (Group Ids) vor. Ein Benutzer kann gleichzeitig mehreren Gruppen angehören. Dies ist dann sinnvoll, wenn dieser an mehreren Projekten beteiligt ist oder auf Daten verschiedener Bereiche Zugriff haben soll. Zugriffsberechtigungen auf Verzeichnisse und Dateien können individuell für Benutzer und Gruppen vergeben werden.
Es gibt auf allen Multiuser-Systemen einen privilegierten Benutzer, der das System verwaltet. Er wird häufig auch als Systemadministrator oder Superuser bezeichnet. Dieser Benutzer hat im Falle von UNIX den Namen *root* und die User Id 0. Für ihn gibt es keine Zugriffsbeschränkungen. Er kann neue Benutzer anlegen und Zugriffsrechte vergeben. Auf die Aufgaben dieses Benutzers wird im Kapitel *Administration* näher eingegangen.

Vom Zentralrechner zum Netzwerk

In den letzten Jahren wurde die Hardware immer billiger und leistungsfähiger. Die Möglichkeiten neuer dezentraler Systeme mit ihren grafischen Oberfächen führten zu einem Rückgang der Akzeptanz von Großrechnern.
Anstelle eines Zentralrechners mit einer großen Anzahl von Terminals und vielen Benutzern findet man heute immer häufiger Arbeitsplätze, an denen jeder Mitarbeiter einen oder mehrere Rechner für sich alleine hat. Diese sind durch ein Netzwerk verbunden und erlauben den einfachen Datenaustausch.
Eine ähnliche Entwicklung hat sich auch im UNIX-Bereich vollzogen: Aus dem rein textorientierten, zentralen UNIX-System wurde eine grafische UNIX-Workstation auf dem Schreibtisch.

virtuelle Terminals

Die Entwicklung, daß ein Benutzer mehrere Rechner zur gleichen Zeit bedient, um auf verschiedene Programme und Daten gleichzeitig zugreifen zu können, spiegelt sich auch in den sogenannten *virtuellen Terminals* wieder. Damit kann sich ein Benutzer mit einem einzigen Bildschirm mehrfach auf dem selben Rechner einloggen, indem er mit einer speziellen Tastenkombination zwischen mehreren virtuellen Terminals umschaltet.

2.2 Multitasking

Neuere Multiuser-Systeme sind in der Regel auch Multitasking-Systeme. Sie bieten die Möglichkeit, viele Aufgaben quasi gleichzeitig abzuarbeiten, was bei einfachen Multiuser-Systemen nicht unbedingt der Fall ist.

Die kleinste Einheit, die ein solches System parallel zu anderen bearbeiten kann wird Prozeß oder auch Task genannt. Bei UNIX, das sowohl ein Multiuser, als auch ein Multitasking System ist, spricht man in der Regel von Prozessen.

Prozesse

Prozesse, die auf einem UNIX-System parallel ablaufen, sind zum Beispiel Programme von verschiedenen Benutzern oder solche, die ständig im Hintergrund ablaufen (Daemons).

Daemons

Ein wesentliches Merkmal von modernen Multitasking-Systemen ist außerdem die Verfügbarkeit von Inter-Prozeß-Kommunikation (IPC). Darunter versteht man Funktionen, die zur Synchronisation oder zum Datenaustausch zwischen Prozessen dienen.

Auf normalen Rechnern, mit nur einem einzigen Prozessor, muß dieser abwechselnd den Prozessen zugeteilt werden, um für den Benutzer eine scheinbar gleichzeitige Verarbeitung zu erreichen.

Diese Aufgabe übernimmt der sogenannte Scheduler. Das ist ein spezieller Prozeß, der eine Liste der normalen Prozesse führt und dafür sorgt, daß der Prozessor in regelmäßigen Abständen den nächsten Prozeß bearbeitet.

Scheduler

Es gibt verschiedene Strategien, nach denen ein Scheduler bestimmen kann, welcher Prozeß als nächstes bearbeitet werden soll. Eine sehr einfache ist, in bestimmten Zeitintervallen (beispielsweise 50ms) den nächsten Prozeß in der Liste auszuwählen, und diesen nach seiner Bearbeitung hinten in die Liste einzureihen (round robin).

Andere Strategien ordnen jedem Prozeß eine Priorität zu, wobei Prozesse mit höherer Priorität mehr Rechenzeit bekommen als Prozesse mit niedrigerer Priorität.

UNIX verwendet zusätzlich sogenannte *Nice Levels*, mit denen der Benutzer die interne Priorität der Prozesse beeinflussen kann. Auf diese Weise kann die Belastung des Systems durch Programme, die im Hintergrund ablaufen, erheblich reduziert

Nice Level

werden. Wichige Prozesse können vom Systemadministrator auch in ihrer Priorität erhöht werden, um eine schnellere Abarbeitung zu gewährleisten.

2.3 Memory-Management

Die Speicherverwaltung eines heutigen UNIX-Systems unterscheidet sich deutlich von der eines einfacheren Betriebssystems wie z.B. MS-DOS. Linux verwendet ein virtuelles Memory Management. Das bedeutet, daß den Programmen mehr Speicher vorgetäuscht wird, als tatsächlich vorhanden ist.

Paging

Das Verfahren, mit dem dies bei Linux ermöglicht wird, nennt sich Paging. Dabei wird mit Hilfe von Tabellen ein großer logischer Adreßraum auf einen kleinen physikalischen abgebildet. Der Speicher, der momentan nicht im physikalischen Hauptspeicher vorhanden ist, wird auf einem sekundären Speichermedium, das ist hier die Festplatte, ausgelagert.

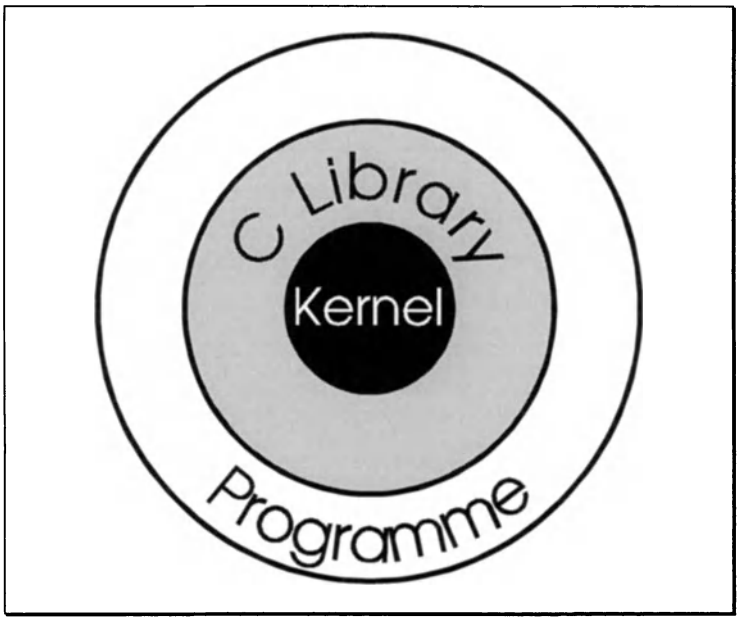
Wird auf eine logische Adresse zugegriffen, die sich gerade auf der Festplatte befindet, so wird der Bereich in den Hauptspeicher geladen, und ein anderer dafür auf der Festplatte abgelegt. Dieser Vorgang geht, wegen den erheblich höheren Zugriffzeit einer Festplatte, natürlich auf Kosten der Geschwindigkeit des Systems.

Swap File/Partition

Um Speicher der Festplatte für die virtuelle Speicherverwaltung und den logischen Hauptspeicher verwenden zu können, müssen auf der Festplatte sogenannte Swap-Files oder Swap-Partitionen eingerichtet werden. Ohne solche Partitionen oder Dateien ist der Hauptspeicher auf die tatsächlich vorhandene Größe beschränkt.

2.4 Schalenmodell

Der Aufbau eines UNIX-Systems wird häufig mit Hilfe eines Schalenmodells dargestellt.



Schematischer Aufbau eines UNIX-Systems

Kernel

Der Kern eines UNIX-Systems wird *Kernel* genannt. Er enthält zum Beispiel den Scheduler und die *Device Driver*. Das sind die Routinen, die den Zugriff auf Interface-Hardware und externe Geräte ermöglichen. Die Speicherverwaltung (Memory Management) befindet ebenfalls im Kernel.

Die Prozesse des Kernels unterscheiden sich von den Prozessen, die in den Schalen um den Kernel ablaufen. Die normalen Prozesse eines Benutzers können jederzeit unterbrochen werden. Sie unterliegen der Steuerung des Schedulers, und ihnen ist ein bestimmter Speicherbereich zugeordnet.
Versucht ein Benutzerprozeß auf den Speicherbereich außerhalb seines eigenen zuzugreifen, so wird er mit der Meldung "segmentation violation" abgebrochen.

UserMode/Kernel Mode

Kernelprozesse dagegen haben universellen Zugriff auf alle Ressourcen des Rechners. Man spricht daher von verschiedenen Modi, in denen Prozesse ablaufen können, dem User Mode und dem Kernel Mode.

Die äußerste Schale des UNIX-Systems besteht aus Programmen, mit denen der Benutzer direkt in Berührung kommt. Dies sind zum einen die Kommando-Shell, über die Betriebssystembefehle abgesetzt werden und zum anderen Anwendungsprogramme, wie eine Textverarbeitung oder eine Datenbank.

Libraries

Zwischen dieser Schale und dem Kernel liegen die verschiedenen Libraries (Bibliotheken), die den Zugriff auf meist in C geschriebene Bibliotheks-Funktionen und auf Routinen des Kernels ermöglichen.

Diese Libraries werden nach dem Compilieren eines Programms normalerweise zu dem Programm gebunden (gelinkt). Das hat zur Folge, daß das Programm danach, neben seinen eigenen Routinen, auch die Routinen der Library enthält.

Shared Libraries

Da der Platzbedarf statisch gelinkter Programme recht hoch ist, greift man heute meist auf sogenannte *Shared Libraries* zurück. Diese bestehen aus zwei Teilen. Ein kleiner Teil, der nur Referenzen auf die Library enthält, wird zum Programm gelinkt. Die eigentliche Library wird erst zur Laufzeit des Programms geladen. Die in einer Shared Library enthaltenen Routinen können dabei auch von mehreren Programmen gleichzeitig genutzt werden, was zusätzlich Speicherplatz einspart.

Ein weiterer Vorteil ist die Möglichkeit, eine Shared Library gegen eine neuere Version auszutauschen, ohne die darauf aufbauenden Programme neu linken zu müssen. Dies ist jedoch nur dann möglich, falls die Routinen der neuen Bibliothek aufrufkompatibel zur alten Version sind.

2.5 Filesysteme

Ein Filesystem dient der Verwaltung der auf einer Festplatte gespeicherten Dateien. Obwohl jedes Computersystem über derartige Mechanismen verfügt, können diese sehr unterschiedlich aussehen.

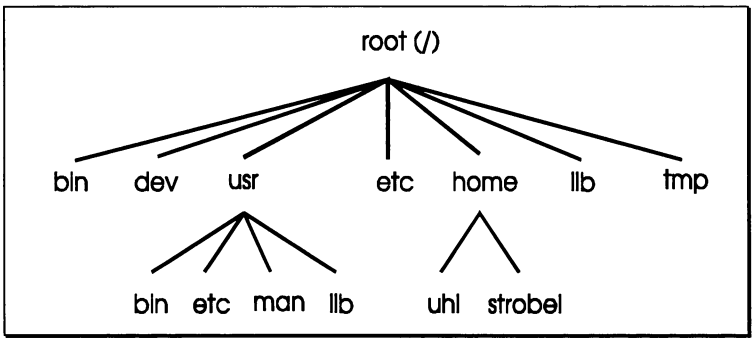
Hierarchische Struktur

Heutige Dateisysteme besitzen eine hierarchische Struktur. Der Benutzer kann seine Dateien auf verschiedene Verzeichnisse verteilen und behält auf diese Weise leichter den Überblick. Der Zugriff auf die einzelnen Dateien erfolgt über sogenannte Pfade.

Im Gegensatz zu MS-DOS wird unter UNIX das /-Zeichen (slash) als Trennsymbol innerhalb eines Pfades benutzt.

Unter MS-DOS werden die Diskettenlaufwerke und die einzelnen Partitionen der Festplatte über einen Buchstaben angesprochen. Unter UNIX sind diese zu einem Filesystem zusammengefaßt und erhalten somit keine getrennten Bezeichnungen. Das bedeutet, daß der Benutzer nicht mehr zwischen den einzelnen Laufwerken und Partitionen unterscheinden kann. Es existiert scheinbar nur ein großes Laufwerk mit einem Dateisystem. Probleme gibt es bei der Verwaltung von Disketten oder anderen Wechselmedien, da diese ja nicht ständig im Laufwerk verbleiben. Sie müssen daher vor dem Zugriff über den Befehl `mount` in das System eingebunden werden, was normalerweise nur vom Systemadministrator durchgeführt werden kann.

mount



UNIX-Dateibaum

Auch die Verwaltung der Dateien und freien Blöcke geschieht unter UNIX in einer anderen Form. MS-DOS erzeugt auf jedem Laufwerk eine sogenannte *File Allocation Table* (FAT), in der die freien und belegten Sektoren festgehalten werden. Ein zweiter Bereich enthält das Wurzel-Verzeichnis. In einem DOS-Unterverzeichnis (Directory) werden neben den Namen der darin enthaltenen Dateien auch deren Attribute, wie Größe oder Datum, gespeichert.

FAT

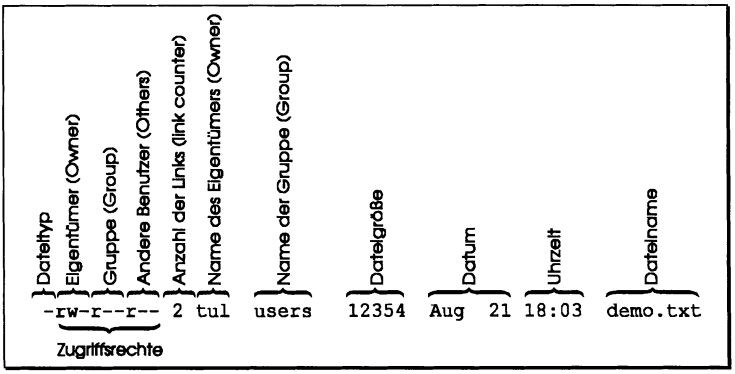
UNIX dagegen legt für jede Datei einen sogenannten i-Node an, in dem die wichtigsten Merkmale, wie Name, Größe, Zugriffsrechte und Startblock gespeichtert sind. Die Verzeichnisse enthal-

i-Node

ten dann nur noch einen Verweis auf einen i-Node. Eine derartige Struktur eignet sich erheblich besser zur Verwaltung größerer Dateisysteme als das FAT-System. Sie ist nicht nur platzsparender, sondern auch effizienter im Zugriff.

Zugriffsrechte

Bei der Erzeugung einer Datei wird unter UNIX nicht nur der Dateiname oder das Datum festgehalten, sondern auch die User-Id des Benutzers, der die Datei erstellt hat bzw. dem sie gehört. Da ein Benutzer auch immer einer Gruppe angehört, wird auch diese gespeichert. Um nun die Daten eines Filesystems vor unerwünschten Zugriffen schützen zu können, werden die Zugriffsrechte für jede Datei getrennt verwaltet. So kann der Zugriff auf den Dateibesitzer oder eine bestimmte Benutzergruppe beschränkt werden. Es ist aber auch möglich, ein allgemeines Zugriffsrecht zu definieren. Unterschieden wird dabei außerdem zwischen Lese-, Schreib- und Ausführungsberechtigung. Diese werden in der Ausgabe des `ls`-Kommando durch die Buchstaben `r`, `w` und `x` gekennzeichnet. Ob es sich dabei um ein Benutzer-, Gruppen- oder Zugriffsrecht für alle handelt, erkennt man an der Position.



Darstellung der Zugriffsrechte durch den Befehl `ls`

Unterverzeichnisse stellen eine Ausnahme dar. Um auf deren Inhalt zugreifen zu können, reicht eine Leseberechtigung. Soll

jedoch in ein Unterverzeichnis verzweigt werden, so muß man sowohl Lese- als auch Ausführungsrechte besitzen.

Links

Eine weitere Besonderheit der UNIX-Dateisysteme ist die Möglichkeit, sogenannte Links zu erzeugen. Soll der Zugriff auf eine Datei von mehreren Stellen des Filesystems aus erfolgen, so könnte man diese einfach kopieren. Natürlich würde auf diese Weise unnötig Speicherplatz verschwendet. In solchen Fällen genügt unter UNIX die Erzeugung eines *Links*.

Hard Link

Dafür gibt es normalerweise zwei Möglichkeiten, nämlich sogenannte Hard und Symbolic Links. Ein Hard Link ist ein zusätzlicher Verweis aus einem Verzeichnis heraus auf eine Datei bzw. deren i-Node. Die Anzahl der Verweise wird über den sogenannten Link Counter verwaltet. Soll eine Datei, auf die mehrere Links verweisen, gelöscht werden, so wird zunächst nur der Link Counter solange erniedrigt, bis er den Wert eins enthält. Erst danach wird die Datei physikalisch gelöscht.

Symbolic Link

Da die Nummern der i-Nodes nur innerhalb eines Filesystems eindeutig sind, können Hard Links nicht dateisystemübergreifend angelegt werden. Eine solche Möglichkeit bieten jedoch die symbolischen Links. Diese können auch auf Verzeichnisse angewandt werden und stellen einen Verweis auf einen Dateinamen dar. Ob dieser wirklich existiert spielt dabei keine Rolle.
Die Unterschiede zwischen den beiden Arten von Links kann man auch gut an der Ausgabe des `ls`-Befehls erkennen:

```
linux1:/etc>ls -l hosts passwd
lrwxrwxrwx 1 tul  users   10 Aug 21 18:05 hosts -> /etc/hosts
-rw-r--r-- 2 root root   863 Aug  9 15:00 passwd
linux1:/etc>
```

Darstellung von Symbolic und Hard Links

Symbolische Links werden mit einem "`l`" in der Spalte für den Dateityp und einem sichtbaren Verweis auf die Originaldatei dargestellt. Hard Links dagegen sind nur an dem erhöhten Link

Counter in der zweiten Spalte (nach den Zugriffsrechten) erkennbar.

Virtuelles Filesystem

Um die Entwicklung verschiedener Filesysteme unter Linux zu erleichtern, wurde eine zusätzliche Schicht, das sogenannte *Virtuelle Filesystem*, zwischen dem Kernel und den eigentlichen Routinen eines Filesystems geschaffen, wie man es auch von kommerziellen UNIX-Versionen kennt. Das virtuelle Filesystem definiert eine Reihe von Routinen, die zum Öffnen, Lesen, Schreiben und Schließen von Dateien benötigt werden und die in jedem Filesystem enthalten sein müssen. Diese eindeutige Schnittstelle ermöglicht erst das problemlose Nebeneinander verschiedener Dateisysteme.

2.6 Geräte

Unter UNIX werden Festplatten ebenso wie Terminals und andere Geräte (Devices) auf eine spezielle Datei im Verzeichnis `/dev` des Dateisystems abgebildet. Der Programmierer kann dadurch auf ein solches Device wie auf eine normale Datei zugreifen.
Diese dateiähnliche Sichtweise von Geräten hat aber auch für den Anwender Vorteile. Soll beispielsweise eine Datei auf einem Drucker, statt auf dem Bildschirm, ausgegeben werden, so genügt es, die Standardausgabe auf das entsprechende Device (`/dev/lp1`) umzulenken.

```
linux2:/home> cat ausgabe.txt >/dev/lp1
```

Einfache Ausgabe auf den Drucker

Auch das Diskettenlaufwerk (`/dev/fd0`), die Maus (`/dev/mouse`), oder die Festplatte werden über einen Eintrag im `/dev`-Verzeichnis angesprochen.

/dev/console	Systemkonsole
/dev/mouse	serielle Maus
/dev/hda	erste AT-Bus Festplatte
/dev/hda1	erste Partition der ersten AT-Bus Festplatte
/dev/hda2	zweite Partition
/dev/hdb	zweite AT-Bus Festplatte
/dev/hdb1	erste Partition der zweiten AT-Bus Festplatte
/dev/sda	erste SCSI-Festplatte
/dev/sdb	zweite SCSI-Festplatte
/dev/lp0	erste Druckerschnittstelle (LPT1)
/dev/null	Null-Device, sämtliche Ausgaben werden verschluckt
/dev/ttyN	Virtuelle Konsolen
/dev/ptyN	Pseudoterminals für Login über ein Netzwerk
/dev/ttySN	Serielle Schnittstellen

Liste der wichtigen Gerätetreiber

Major/Minor Numbers

Die Zuordnung der Dateien im Verzeichnis /dev und den entsprechenden Geräten erfolgt über zwei Zahlen, den sogenannten Major und Minor Numbers. Diese bilden auch die Schnittstelle zum Kernel. Außerdem unterscheidet man bei Geräten zwischen Character- und Blockdevices. Erstere arbeiten zeichenorientiert und werden daher in erster Linie für Geräte wie Terminal oder serielle Schnittstellen benutzt. Letztere dienen vor allem zur Übertragung größerer Datenblöcke bei Festplatten oder anderen Datenträgern.

```
linux1:/dev>ls -l
                                ...
brw-r-----   1 root     root       3,   0 Aug 29  1992 hda
brw-r-----   1 root     root       3,   1 Aug 29  1992 hda1
brw-r-----   1 root     root       3,   2 Aug 29  1992 hda2
                                ...
crw-rw-rw-   1 root     root       4,   0 Aug 16 12:26 tty0
crw--w--w-   1 tul      users      4,   1 Aug 21 15:15 tty1
                                ...
linux1:/dev>
```

Pseudodateien im Verzeichnis /dev

Die Major-Device-Number (fünfte Spalte) identifiziert den Typ des Gerätes. Auch diese werden, wie man obiger Abbildung entnehmen kann, durch den Befehl `ls` angezeigt. Normalerweise existiert für jede Major-Number ein eigener Device-Treiber im Kernel. Werden mehrere Geräte vom gleichen Typ angeschlossen, so werden diese durch ihre Minor Device Number (sechste Spalte) voneinander unterschieden und vom selben Treiber bedient.

2.7 Shells

Die UNIX-Befehle wie `ls`, `cd` oder `man`, die man in der normalen Kommandozeile eingibt, werden nicht vom UNIX-Betriebssystem selbst ausgeführt, sondern es sind Programme, die meist in den Verzeichnissen `/bin` und `/usr/bin` stehen. Bei der Eingabe solcher Befehle, befindet man sich nicht im direkten Kontakt mit dem UNIX-Betriebssystem, sondern nur in der äußersten Schale, einem Programm, das Shell genannt wird.

Allgemeines

Die Aufgaben der Shell sind in etwa mit denen des `command.com` unter DOS vergleichbar, wenngleich eine UNIX-Shell wesentlich mehr Möglichkeiten bietet als der DOS-Kommandointerpreter.
Die Shell kennt nur wenige eigene Befehle und verbringt die meiste Zeit damit, eine Eingabe des Benutzers von der Standard-Eingabe, also der Tastatur zu lesen, und dann das entsprechende Programm zu starten. Bei der Eingabe von `ls -l` ruft die Shell beispielsweise das Programm `/bin/ls` mit der Option `-l` auf, und es wird der Inhalt des aktuellen Verzeichnisses auf dem Bildschirm ausgegeben.

Pipes, Shell Scripts

Daneben ist eine Shell unter anderem in der Lage, Ein- und Ausgaben umzuleiten, mehrere Befehle über sogenannte *Pipes* zu verketten oder Shell Scripts auszuführen. Ein Shell Script besteht aus einer beliebigen Anzahl von Anweisungen, wie sie auch interaktiv ausgeführt werden könnten. Über Pipes kann die

Ausgabe eines Kommandos als Eingabe eines zweiten Befehls benutzt werden. Von dieser Möglichkeit wird sehr häufig Gebrauch gemacht, falls eine längere Bildschirmausgabe über das Kommando `more` seitenweise angezeigt werden soll.

```
linux1:/home/tul> ls -l | more
```

Anwendung einer Pipe

Da auch eine Shell ein gewöhnliches Programm ist, das beim Einloggen eines Benutzers gestartet wird (Login Shell), kann diese gegen andere Versionen ausgetauscht werden.

Standard Shells

Auf den meisten kommerziellen UNIX-Systemen findet man drei verschiedene Typen von Shells. Eine Bourne-Shell (`sh`), eine Korn-Shell (`ksh`) und eine C-Shell (`csh`). sh, ksh, csh
Die Bourne-Shell war die erste Shell unter UNIX und bietet daher auch keinen großen Komfort. Die Korn-Shell ist eine Erweiterung der Bourne-Shell und wird bei kommerziellen Systemen relativ häufig verwendet.

Die C-Shell entstand wie das BSD-UNIX an der Berkeley University und bietet im Gegensatz zur Bourne- und Korn-Shell eine einfache History-Funktion, sodaß auf schon ausgeführte Kommandos zugegriffen werden kann. Die sogenannte *Alias Substitution* erlaubt die automatische Ersetzung bestimmter Kommandos, die in einer der Initialisierungsdateien oder interaktiv festgelegt wurden. Aufgrund der C-ähnlichen Syntax ist diese Shell-Variante besonders bei Programmierern beliebt.
Die C-Shell führt beim Start als Login Shell die Kommandos der Datei `.login` aus, die sich zu diesem Zweck im Home-Verzeichnis befinden muß. Da ein Benutzer normalerweise nur eine aktive Login Shell besitzt, werden die darin enthaltenen Befehle auch nur einmal beim Login abgearbeitet. .login, .cshrc
Ein Shell Script, das im Gegensatz dazu, beim Start jeder neuen C-Shell ausgeführt wird, ist die Datei `.cshrc`, die ebenfalls im Home-Verzeichnis gesucht wird. Häufig werden in dieser Datei

Pfade gesetzt, Environment-Variablen definiert und Aliassubstitutionen deklariert.

Alias

Ein Alias ist ein zusätzlicher Name für einen Befehl, bei dem auch Optionen angegeben werden können. Der folgende Ausschnitt einer `.cshrc` Datei legt zum Beispiel den Alias `l` für den Befehl `ls` mit der Option `-lF` an und ändert den `ls`-Befehl, so daß automatisch die Option `-F` verwendet wird.
Diese fügt in der Anzeige bei Verzeichnissen ein / und bei ausführbaren Dateien einen * an den Namen an.

```
PS1='\h:$PWD>'
alias l "ls -lF"
alias ls "ls -F"
alias rm "rm -i"
```

Ausschnitt aus einer .cshrc Datei

Der Alias für `rm` bewirkt, daß vor dem Löschen einer Datei nachgefragt wird, ob der Benutzer die Datei auch wirklich löschen möchte.
Da dieses Verhalten beim Löschen mehrerer Dateien auch störend sein kann, ist es möglich, entweder den `rm`-Befehl direkt durch Voranstellen seines Pfades aufrufen (`/bin/rm`) oder mit dem Kommando `unalias rm` den Alias zu löschen.

```
dirk1:/home/stefan# ls
Mail/          faqxp4.apr     test*
dirk1:/home/stefan# /usr/bin/ls
Mail           faqxp4.apr     test
dirk1:/home/stefan#
```

Auswirkung des Alias für ls und Umgehen des Alias durch den Pfad

Alternative Shells
bash, tcsh

Unter Linux wird in der Regel keine der oben genannten Shells verwendet. Statt dessen kommen erweiterte Varianten dieser Shells zum Einsatz. Diese frei erhältlichen und komfortableren Versionen sind für fast alle UNIX-Systeme verfügbar und machen die originalen Shells überflüssig. Die Bourne Shell wird dabei durch die sogenannte *Bourne Again Shell* (`bash`) und die C-Shell durch die `tcsh` ersetzt.

2.8 Daemons

Daemons sind spezielle Prozesse, die im Hintergrund ablaufen und meist wichtige Aufgaben innerhalb eines UNIX-Systems übernehmen. Große Teile des Betriebssystems laufen somit als eigenständige Programme. Auf diese Weise kann der Betriebssystemkern relativ klein gehalten werden. Außerdem lassen sich, auch während des Betriebs, einzelne Daemons aktivieren oder nach einer Änderung in der Konfiguration neu starten. Da Daemons als eigenständige Prozesse laufen, können diese auch parallel nebeneinander arbeiten und blockieren somit keine anderen Programme.

Printer Daemon (lpd)

Der Line Printer Daemon (`lpd`) überprüft in regelmäßigen Abständen das Verzeichnis `/usr/spool` auf neue Druckaufträge und gibt diese auf dem ensprechenden Drucker aus. Zur Ausgabe einer Datei auf einen Drucker steht unter Linux das vom Berkeley-System bekannte `lpr`-Kommando zur Verfügung. Neue Druckaufträge werden normalerweise immer an das Ende der Warteschlange angehängt, ehe sie vom Drucker Daemon ausgegeben werden.

Cron Daemon

Wünscht ein Benutzer die Ausführung eines Programmes zu bestimmten Zeitpunkten oder in regelmäßigen Abständen, so kann er dies über den sogenannten Cron-Daemon erreichen. Dieser verwaltet für jeden Benutzer eine eigene Tabelle, in der die einzelnen Zeitpunkte eingetragen werden, an denen die gewünschten Prozesse starten sollen. Die Ausgabe eines ausgeführten Kommandos oder ensprechende Fehlermeldungen werden dem Benutzer als Mail zugeschickt. Soll ein Script nur einmal zu einem bestimmten Zeitpunkt ausgeführt werden, so sollte man das Kommando `at` benutzen. Für regelmäßige Aufgaben ist ein Eintrag in die Tabelle des Cron-Daemons (`crontab`) nötig. Zu diesem

crontab

Zweck existiert übrigens ein eigenes Kommando namens `crontab`.

Internet Daemon (inetd)

Die meisten Internet-Dienste sind ebenfalls als Daemons implementiert. Sie werden nur dann aktiviert, wenn tatsächlich eine Anforderung des entsprechenden Dienstes von einem anderen Rechner vorliegt. Wären sie wie andere Daemons ständig im Hintergrund aktiv, so würden sie nur unnötig Speicherplatz und Rechenzeit verbrauchen. Daher werden diese Daemons nicht gleich beim Booten des Systems gestartet.

Portnummer

Ein UNIX-System verfügt normalerweise über einen sogenannten *Internet Daemon* (Internet Super Server), der auf eventuelle Anfragen aus dem Netz wartet. Jeder Dienst besitzt eine feste Portnummer. Über die Datei (`/etc/services`) erfolgt die Zuordnung zwischen einer Portnummer und dem jeweiligen Dienst . In einer weiteren Datei (`/etc/inetd.conf`) sind dann die entsprechenden Daemons verzeichnet, die den gewünschten Service bieten. Erst wenn tatsächlich ein Verbindungswunsch vorliegt, startet `inetd` den entsprechenden Daemon, der die Verbindung übernimmt. Nach Beendigung der Verbindung wird der Daemon ebenfalls beendet, `inetd` hingegen läuft weiter.

Syslog Daemon

Da ein Daemon üblicherweise keine direkten Ausgaben auf den Bildschirm macht, wurde ein eigener Protokoll-Daemon geschaffen, der Ausgaben und Fehlermeldungen anderer Daemons aufnimmt. Diese können dann auf die Konsole ausgegeben, in Dateien geschrieben oder als Mail an den Systemadministrator weitergeleitet werden.

Netzwerk-Daemons

Wie bereits erwähnt, wurden die meisten Internet-Dienste, wie `ftp`, `telnet` oder `mail`, über separate Daemons realisiert, die

von `inetd` bei Bedarf aktiviert werden. Die folgende Aufzählung gibt einen Überblick über die wichtigsten unter Linux vorhandenen Netzwerk-Daemons.

- **`bootpd`** - wird zum Booten von Diskless Workstations und X-Terminal benötigt.
- **`fingerd`** - ermöglicht die Anfrage (`finger`), welche Benutzer auf einem (anderen) System aktiv sind.
- **`ftpd`** - dient zur Dateiübertragung mit `ftp` von einem auf ein anderes System.
- **`mountd`** - erlaubt die Einbindung eines Dateisystems eines anderen Rechners in das lokale Filesystem.
- **`nfsd`** - stellt Daten als NFS Server zur Verfügung.
- **`nntpd`** - übermittelt News aus dem USENET.
- **`rlogind`** - ermöglicht das Einloggen von einem anderen System aus mittels eines `rlogin`-Kommandos.
- **`rshd`** - erlaubt die Ausführung eines Kommandos von einem anderen System aus.
- **`smail`** - verschickt und empfängt Mails im Netz.
- **`talkd`** - ermöglicht die interaktive Kommunikation mit anderen Benutzern über das Kommando `talk`.
- **`telnetd`** - ermöglicht, ähnlich wie `rlogind`, das Einloggen eines Benutzers von einem anderen Rechner aus.
- **`tftpd`** - wird wie `bootpd` zum Booten anderer Maschinen im Netz benutzt.

2.9 Befehlsübersicht

Um den Einstieg in die Bedienung von Linux zu erleichtern, sollen abschließend die wichtigsten UNIX-Kommandos aufgezählt und kurz erläutert werden. Nähere Infomationen zu den einzelnen Befehlen können der gängigen UNIX-Literatur oder dem Online Manual entnommen werden.

- **ls** - gibt eine Liste von Dateien und Verzeichnissen aus. Zusätzlich können das Erstellungsdatum, die Dateigröße, die Zugriffsrechte und der Besitzer angezeigt werden. Auch die rekursive Ausgabe ganzer Verzeichnisbäume ist optional möglich.
- **cd** - wechselt in ein anderes Verzeichnis. Wird kein Parameter angegeben, so befindet man sich anschließend im eigenen Home-Verzeichnis.
- **cp** - kopiert die übergebenen Dateien von einem in ein anderes Verzeichnis. Optional kann auch ein ganzer Verzeichnisbaum rekursiv kopiert werden.
- **mv** - verschiebt eine Datei innerhalb eines Dateisystems. Kann auch zum Umbenennen einer Datei oder eines Verzeichnisses benutzt werden.
- **rm** - entfernt eine Datei. Optional kann auch ein ganzer Dateibaum rekursiv gelöscht werden.
- **mkdir** - erzeugt ein neues Unterverzeichnis.
- **rmdir** - entfernt ein leeres Unterverzeichnis.
- **exit** - verläßt die aktuelle Shell.
- **more** - zeigt den Inhalt einer Textdatei seitenweise auf dem Bildschirm an. Außerdem können innerhalb der Datei Begriffe gesucht werden.
- **man** - zeigt die Online-Dokumentation (Manual pages) zu einem übergebenen Befehl an.
- **cat** - dient eigentlich zum Aneinanderhängen von Textdateien, kann aber auch zur Ausgabe einer Datei benutzt werden.
- **grep** - sucht innerhalb der übergebenen Datei nach einem beliebigen Muster.
- **passwd** - ändert das Passwort eines Benutzers.
- **ps** - listet die laufenden Prozesse mit ihrer Prozeß-Id auf.
- **kill** - beendet einen Prozeß anhand der übergebenen Prozeß-Id.
- **su** - wechselt die User Id temporär, ohne nochmals einen Login durchführen zu müssen. Wird als zusätzlicher Parameter "-" übergeben, so entspricht dies einem erneuten Login.

Vernetzung

Ein wesentlicher Aspekt bei der Diskussion um heutige Workstations und deren Betriebssysteme ist die Netzwerkfähigkeit, also die Möglichkeiten zur Integration der Systeme in bestehende Netze.
Die gesamte Entwicklung von Linux wäre ohne das Internet nicht möglich gewesen. Daher existierte auch schon in sehr frühen Versionen des Kernels die Einbindung des TCP/IP-Protokolls mit entsprechenden Treibern für PC-Netzwerkkarten.
In diesem Kapitel werden neben einigen Grundlagen zur Netzwerkthematik die Programme beschrieben, mit denen von Linux und anderen UNIX-Versionen auf Netzwerkdienste zugegriffen werden kann.

3.1 Netzwerk-Hardware

Die Hardwareanforderungen für einen Netzanschluß sind recht gering. Falls man beispielsweise zwei Rechner besitzt, kann man mit zwei einfachen Ethernetkarten, einem Stück Thin-Ethernet-Kabel, zwei T-Stücken und zwei Abschlußwiderständen ein einfaches Netzwerk (LAN) aufbauen. Die Kosten für diese zusätzliche Hardware liegen unter 300 DM liegen, was die Vernetzung auch für den Privatanwender erschwinglich macht.

LAN

Noch preiswerter kann man seinen Rechner über eine SLIP-Verbindung an ein Netzwerk anschließen. SLIP steht für Serial Line Interface Protokoll und ermöglicht eine TCP/IP-Verbindung über ein serielles Kabel oder ein Modem. Das Gegenstück einer solchen SLIP-Verbindung ist meist eine größere Workstation

SLIP

oder ein SLIP Router, der die IP-Pakete zu einem Ethernet weiterleitet.
Eine derartige Verbindung ist besonders bei Studenten oder Mitarbeitern an Hochschulen beliebt, die einen SLIP Router zur Verfügung haben und sich so über ein Modem direkt mit dem Netzwerk der Hochschule und dem Internet verbinden können.
Da die Übertragung von IP-Paketen über SLIP nicht sehr effizient ist, wurde ein weiteres Protokoll namens PPP geschaffen. Die Linux-Treiber befinden sich momentan noch im Teststadium, dürften jedoch bald standardmäßig im Kernel enthalten sein.

PPP

Auch über die parallele Schnittstelle und einem Protokoll namens PLIP kann eine preiswerte TCP/IP-Anbindung erfolgen. Dazu werden die beiden Rechner mit einem sogenannten Null-Printer-Kabel über eine freie parallele Schnittstelle verbunden. Besonders interessant dürfte diese Art von Netzwerk für den Datentransfer zwischen Notebook und Workstation sein.

PLIP

3.2 TCP/IP

Der de facto Standard für die Vernetzung von UNIX-Rechnern ist TCP/IP, ein Protokoll, das für das Internet entwickelt wurde und inzwischen für fast alle Rechnerplattformen verfügbar ist.

TCP/IP ist keine Norm wie die vielen Standards von ANSI, ISO oder IEEE, sondern eine herstellerunabhängige Definition, die in Form eines sogenannten RFC jedermann zugänglich ist.
RFC steht für Request for Comment und ist meist die Beschreibung eines Protokolls oder ein Vorschlag für ein neues Protokoll. Die RFCs sind auf vielen FTP-Servern oder von einem Network Information Center (NIC), z.B. `nic.ddn.mil` erhältlich.

RFC

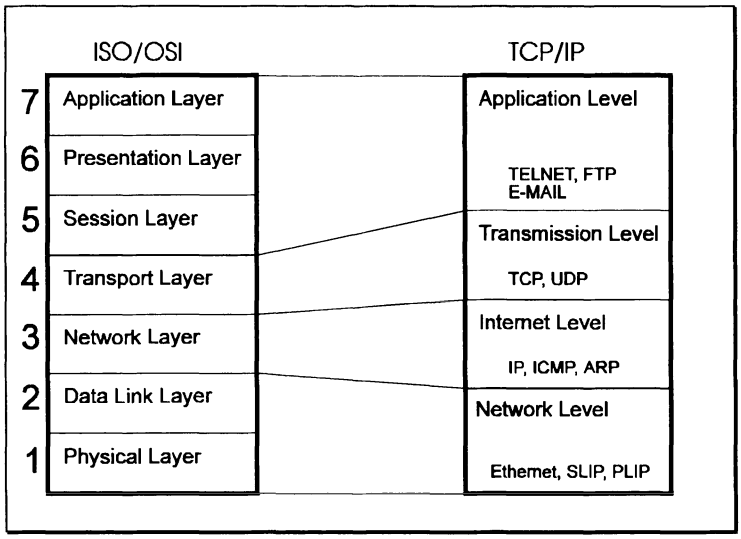
TCP/IP entstand Anfang der 70er Jahre als neues Protokoll für das ARPANET (später DARPANET und dann Internet), welches damals hauptsächlich die amerikanischen Universitäten untereinander verband.
Finanziert wurde dieses Netz von der Advanced Research Project Agency, einer Institution der amerikanischen Regierung, die sonst vor allem militärische Projekte durchführte. Es ist daher

nicht verwunderlich, daß TCP/IP auch vom amerikanischen Verteidigungsministerium zum Standard erklärt wurde.
Inzwischen besteht das Internet weltweit aus vielen Teilnetzen. So sind die großen Hochschulen in Baden Württemberg meist über das Landesforschungsnetz BelWü am Internet angeschlossen. Für Interessenten außerhalb des universitären Bereiches besteht die Möglichkeit von kommerziellen Internet-Anbietern wie *XLink* in Karlsruhe oder der *EUnet GmbH* in Dortmund Anschluß an das Internet zu bekommen.

Das Internet

Aufbau

TCP/IP besteht im wesentlichen aus 4 Ebenen, die sich mit geringen Abweichungen dem heute üblichen ISO/OSI-Referenzmodell zuordnen lassen.



Gegenüberstellung der ISO/OSI- und TCP/IP-Schichten

Die unterste Ebene wird als Netzwerkebene bezeichnet, und entspricht ungefähr den Ebenen eins und zwei des ISO/OSI-Referenzmodells. In der Regel wird auf dieser Ebene Ethernet verwendet.

Es sind zwar auch andere Verbindungen wie zum Beispiel Token Ring möglich, unter Linux sind derzeit jedoch nur Treiber für Ethernetkarten, SLIP und PLIP verfügbar.

IP

Die zweite Ebene ist die Internet Ebene mit dem Internet Protocol (IP). Sie entspricht annähernd der Ebene drei nach ISO/OSI. Hier können Datenpakete (Datagramme) unabhängig von dem eigentlichen Netzwerk, das die Verbindung ermöglicht, übermittelt werden. Die Adressierung eines Rechners erfolgt über die sogenannte IP-Adressen, die jeweils für genau eine Netzwerkschnittstelle steht.

TCP und UDP

Die dritte Ebene entspricht der Ebene vier nach ISO/OSI und bietet TCP und UDP als Protokolle. TCP steht für *Transmission Control Protocol* und stellt eine virtuelle Verbindung her, während UDP *User Datagram Protocol* bedeutet und einen einfachen Paketdienst bietet. Diese Datagramme sind, im Gegensatz zu einer Verbindung auf der IP-Ebene, nicht an den speziellen Aufbau von IP-Paketen gebunden.

Portnummern

Neben der Netzwerkadresse wird zum Aufbau einer Verbindung auf dieser Ebene auch eine sogenannte Portnummer benötigt, der meist ein bestimmter Netzwerkdienst (in `/etc/services`) zugeordnet ist. Auf diese Weise sind auch mehrere voneinander unabhängige Verbindungen über eine Schnittstelle möglich.

Von diesen Portnummern sind einige für die Standard TCP/IP Applikationen reserviert. Für `telnet` ist dies zum Beispiel der Port 23, für `ftp` Port Nummer 20 und 21.

Sockets

Um eine einheitliche Schnittstelle zur Programmierung von Netzwerkapplikationen zur Verfügung zu stellen, führten die Entwickler des BSD-UNIX Anfang der achziger Jahre die sogenannten *Berkeley-Sockets* ein. Dabei handelt es sich eine Reihe von Kernel-Routinen, die zum Aufbau einer Verbindung zwischen zwei Rechnern und zur Übertragung von Daten benötigt werden. Über Sockets können sowohl TCP- als auch UDP-Verbindungen realisiert werden.

LAN/WAN

Bemerkenswert ist, daß es keine Rolle spielt, ob die Datenübertragung innerhalb eines lokalen Netzwerkes (LAN) oder

über ein weltumspannendes Wide Area Network (WAN) erfolgt. Die Programmierschnittstelle ist immer dieselbe.
Auch der Linux Kernel besitzt eine Socket-Schnittstelle, so daß die meisten bekannten UNIX-Netzwerkanwendungen zur Verfügung stehen.

Anwendungen

Die vierte Ebene von TCP/IP deckt die ISO/OSI-Schichten fünf bis sieben ab und wird als Applikationsebene bezeichnet. Während die Ebenen eins bis drei in der Regel vom Betriebssystem abgedeckt werden, bei Linux sind es Teile des Kernels, besteht die Ebene vier aus gewöhnlichen Programmen.
Die wichtigsten Dienste auf dieser Ebene sind Telnet für das virtuelle Terminal, FTP um Dateien zu transferieren und E-Mail zur Übertragung elektronischer Briefe.
FTP und Telnet existieren als gleichnamige Programme, die vom Benutzer direkt verwendet werden können. E-Mail wird dagegen von der TCP/IP-Seite nur durch einen einfachen Transportmechanismus unterstützt, der von verschiedenen anderen Programmen verwendet wird.
Um zum Beispiel mit FTP eine Verbindung zum wichtigsten Linux FTP-Server in Finnland aufzubauen, gibt man `ftp nic.funet.fi` ein. Damit wird das FTP-Programm gestartet, das dann versucht, über die fest vereinbarte Portnummer 21 des `FTP`-Daemons eine TCP-Verbindung zum anderen Rechner aufzubauen.

FTP

Ist diese Verbindung aufgebaut, so wird man aufgefordert, seine User Id anzugeben. Bei FTP-Servern die frei zugänglich sind, kann man den Benutzernamen ftp oder anonymous verwenden und sollte danach die eigene E-Mail-Adresse als Passwort angeben.

```
linux2:/home/stefan>ftp sun1
Connected to sun1.
220 sun1 FTP server (SunOS 4.1) ready.
Name (sun1:stefan): ftp
331 Guest login ok, send ident as password.
Password (sun1:ftp): stefan@linux2.rz.fh-heilbronn.de
230 Guest login ok, access restrictions apply.
ftp> ls
200 PORT command successful.
150 ASCII data connection for /bin/ls (141.7.1.41,1157) (0
bytes).
total 6
drwxrwxrwx  2 0        150           512 Jul 16 16:14 Incoming
-rw-r--r--  1 0        1             139 Aug 22 12:33 README
drwxr-xr-x  2 0        150           512 May 10 09:51 bin
drwxr-xr-x  2 0        150           512 May 10 09:54 dev
drwxr-xr-x  5 0        150           512 Jun 20 15:36 pub
drwxr-xr-x  3 0        150           512 May 10 09:52 usr
226 ASCII Transfer complete.
ftp> get README
200 PORT command successful.
150 ASCII data connection for README (141.7.1.41,1158) (139
bytes).
226 ASCII Transfer complete.
142 bytes received in 0 seconds (0.14 Kbytes/s)
ftp> bye
221 Goodbye.
```

Beispiel einer FTP-Session

Danach kann man mit den Befehlen des FTP-Programms wie `cd`, `dir`, `get` oder `put` Programme suchen und übertragen. Diese Befehle entsprechen nicht den Kommandos, die vom FTP-Protokoll verwendet werden, sondern sie werden vom `FTP`-Programm erkannt und in entsprechende Protokollbefehle, zum Beispiel `PORT` oder `RETR` umgesetzt. Nur diese Protokollbefehle werden übertragen. Für die Datenverbindung bei der Übertragung von Dateien wird eine zweite TCP-Verbindung hergestellt.
Die einzelnen Kommandos entnimmt man am besten der Manual page zu `ftp` oder aus der Hilfe innerhalb des `FTP`-Programms, die man durch Eingabe von `help` bekommt.

Telnet

Um sich mit Telnet auf einem anderen Rechner einzuloggen, ruft man das Programm mit einem Rechnernamen oder einer IP-Adresse auf. Dabei wird eine TCP-Verbindung zu dem Telnet Daemon des angegebenen Rechners aufgebaut.

```
linux2:/home/stefan>telnet sun1
Trying 141.7.1.20...
Connected to sun1.
Escape character is '^]'.


SunOS UNIX (sun1)

login:
```

Telnet Login

Das Telnet-Programm verwendet die Portnummer des Telnet Daemons nur als Vorbelegung. Beim Aufruf von Telnet kann nach dem Hostnamen optional eine Portnummer angegeben werden.
So kann man zum Beispiel mit `telnet <Hostname> 25` eine Verbindung zum Port 25 eines Rechners aufbauen, der für SMTP, also das E-Mail-Protokoll reserviert ist, oder mit der entsprechenden Portnummer eine Verbindung zum News Daemon eines News Servers (Port 119) herstellen.
Diese Möglichkeit geht sogar so weit, daß man durch Eingabe von `help` von den Server Daemons eine Beschreibung ihrer Protokollbefehle bekommt. Bei der Fehlersuche sind derartige Features eine wertvolle Hilfe.
Darüberhinaus gibt es Server, die auf bestimmten Ports Spiele wie Textadventures, sogenannte Multi User Dungeons (MUDs) oder Schach anbieten. Dazu muß man nur den Namen des Hosts und die Portnummer kennen.

MUDs und Chess Server

Den Internet Chess Server (ICS) findet man zum Beispiel auf dem Host `cirrus.gp.cs.cmu.edu` unter der Portnummer 5000. Um diesen Server zu benutzen, reicht es aus, mit

```
linux1:/home/tul> telnet cirrus.gp.cs.cmu.edu 5000
```

eine Verbindung herzustellen.
Das Programm `xboard`, das normalerweise als grafisches Frontend für das GNU-Schach-Programm verwendet wird, ist zum Beispiel in der Lage, eine solche Telnet-Verbindung aufzubauen und parallel eine Stellung grafisch darzustellen.
Eine gute Quelle für derartige Internet-Dienste sind Listen, die meist unter dem Namen `internet.services` oder als `internet.ressources` auf verschiedenen FTP-Servern zu

finden sind. Genauso wie FAQs (Frequently asked Questions) werden diese Listen auch regelmäßig in der Newsgruppe `news.answers` veröffentlicht. Es lohnt sich also gerade für Neueinsteiger. sich in die Bedienung eines Newsreaders einzuarbeiten, da dadurch der Zugang zu anderen Servern und Informationsquellen sehr viel einfacher wird.

E-Mail

Der TCP/IP E-Mail-Dienst besteht in der Regel aus einem Daemon, der die Übertragung der Nachrichten zu anderen Rechnern mit dem SMTP-Protokoll übernimmt, und Programmen, mit denen man Briefe lesen und schreiben kann.
Unter Linux sind für die Übertragung sowohl das weit verbreitete `sendmail`-Programm als auch Varianten wie `smail` verfügbar. Als sogenanntes Mail Frontend zum Lesen und Schreiben von Briefen ist in den meisten Paketen `elm` enthalten. Außerdem existieren grafische Programme, wie zum Beispiel *MuMail*, mit dem Briefe komfortabel unter X11 verwaltet werden können. (siehe auch im Kapitel *Netzanwendungen*)

3.3 Berkeley r-Utilities

Einen großen Einfluß auf die Netzwerkmöglichkeiten unter UNIX hatte das an der Universität von Berkeley entwickelte BSD-UNIX. Durch die gelungene Integration von TCP/IP in das UNIX- Betriebssystem trug diese Implementierung wesentlich zur weiteren Verbreitung von TCP/IP bei.
Als Berkeley r-Utilities bezeichnet man heute eine Reihe von Programmen, deren Name jeweils mit einem r beginnt. Die wichtigsten sind `rlogin`, `rsh` und `rcp`, wobei das r für Remote steht.
Die Berkeley r-Utilities gehören inzwischen zum Standard einer vernetzten UNIX-Workstation und sind im Laufe der Zeit um einige Programme wie zum Beispiel `rwho` oder `ruptime` erweitert worden.
Unter Linux wurden, wie auch in vielen kommerziellen UNIX-Varianten, die Utilities aus der Berkeley 4.2 Distribution portiert.

Die Grundidee dieser Utilities ist es, einem Benutzer, der auf mehreren Rechnern im Netz einen Account (Zugang) besitzt, eine einfache Möglichkeit zu geben, sich auf einem anderen Rechner im Netzwerk einzuloggen, Programme ablaufen zu lassen, oder Dateien zu kopieren, ohne daß er sich dazu jedesmal mit seinem Passwort anmelden muß.

"trusted users"

Damit dies möglich ist, und dennoch keine Sicherheitslücke entsteht, kann definiert werden, welchen Benutzern von anderen Rechnern vertraut wird. Nur diese Benutzer können sich dann auf dem Rechner anmelden und Dienste des Rechners verwenden, ohne ein Passwort anzugeben. Diese Definition geschieht systemweit in der Datei `/etc/hosts.equiv` und in der Datei `.rhosts` im Home-Verzeichnis des jeweiligen Benutzers für seinen eigenen Account.

Für einen weiteren Sicherheitsgewinn sorgt die Verwendung von sogenannten privilegierten Ports. Dabei überprüft der Server die TCP-Portnummer des Clients. Ist diese kein privilegierter Port, so wird die Verbindung verweigert.

privilegierte Portnummern

Privilegierte Ports stehen auf UNIX-Rechnern nur Benutzern mit Superuser-Berechtigung zur Verfügung. Dadurch wird verhindert, daß ein normaler Benutzer mit einem selbstgeschriebenen anderen Programm einen falschen Rechner- oder Benutzernamen vortäuscht, um sich so Zugang zu einem Rechner zu verschaffen. Dies bedeutet natürlich, daß die r-Utilities mit der User Id *root* ablaufen müssen.

Kerberos

Neuere Implementationen der Berkeley r-Utilities unterstützen auch *Kerberos,* um eine weitere Steigerung der Sicherheit zu erreichen. Kerberos ist ein System zur Überprüfung von Passwörtern und Benutzerkennungen, das am *Massachusetts Institute of Technology* (MIT) entwickelt wurde.

rlogin

Das Programm `rlogin` funktioniert für den Benutzer sehr ähnlich wie das `telnet`-Programm, jedoch muß man bei entsprechender Konfiguration keine Benutzerkennung und kein Passwort eingeben. Die Angabe einer abweichenden Portnummer ist bei `rlogin` nicht möglich.

rcp

Mit `rcp` können Dateien zwischen verschiedenen Rechnern kopiert werden. Dazu muß jedoch der genaue Zugriffspfad angegeben werden. Daher wird in vielen Fällen das interaktive FTP Utility oder der Zugriff über ein *Network File System* (NFS) bevorzugt. Im Gegensatz zum normalen FTP können über `rcp` auch Dateihierarchien rekursiv kopiert werden.

rsh

Die *Remote Shell* `rsh` wird dazu benutzt, Programme auf anderen Rechnern zu starten. Neben dem auszuführenden Kommando wird der Name eines Rechners und optional eine Benutzerkennung angegeben. Die Ausgabe des ausgeführten Befehls wird dann über das Netz auf die lokale Maschine umgelenkt.

Datenübertragung mit rsh

Dieses Programm eignet sich neben der Ausführung von Kommandos auf einem anderen Rechner auch dazu, schnell Daten zwischen verschiedenen Rechnern zu übertragen. Dabei nutzt man die Eigenschaft von `rsh` aus, die eigene Standardein- und -ausgabe mit dem Programm, das auf dem Zielrechner gestartet wird, zu verbinden. Auf diese Weise kann beispielsweise die Sicherung von Dateien einer lokalen Festplatte auf ein Bandlaufwerk (streamer) eines anderen Rechners im Netz erfolgen.

3.4 NFS

Das *Network File System* (NFS) macht es möglich Dateisysteme, die von anderen Rechnern dazu freigegeben (exportiert) wurden, als Teil des eigenen Filesystems zu benutzen (mounten). Damit erhält man einen transparenten Zugriff auf Verzeichnisse eines anderen Rechners.

Das folgende Beispiel zeigt den Zugriff auf Dateien im `/home`-Verzeichnis eines anderen Rechners (`stef1`) im Netzwerk. Zunächst ist das Verzeichnis `/stef1` auf Maschine `dirk1` leer.

Nach dem Mount-Vorgang entählt dieses quasi alle Dateien des Verzeichnisses `/home` des Rechners `stef1`.

```
dirk1:/# ls
bin/          install/      lost+found/  stef1/       var@
dev/          lastlog@      mnt/         tmp/         vmlinux
etc/          lib/          proc/        user/        vmlinux.old
home/         linux@        root/        usr/
dirk1:/# ls stef1
dirk1:/# mount stef1:/home /stef1
dirk1:/# ls stef1
dirk/    fritz/   ftp/     peter/   root/    stefan/
dirk1:/#
```

Beispiel für einen NFS Mount-Vorgang

NFS wurde von Sun entwickelt, und da die Definitionen des Protokolls freigegeben wurden, konnte NFS von vielen anderen Herstellern in ihre Betriebssysteme integriert werden. So wurde es zu einem Standard, der zwar von keiner übergeordneten Instanz kontrolliert wird, sich aber dennoch auf fast allen Plattformen durchgesetzt hat. NFS existiert für fast alle UNIX-Versionen, MS-DOS und auch für andere Betriebssysteme.

Ein wesentliches Merkmal von NFS ist der sogenannte *stateless Server*. Das bedeutet, daß der NFS Server, der Verzeichnisse exportiert, keine Zustandsinformation der Clients speichert, sondern nur einfache Lese- und Schreiboperationen durchführt.

stateless Server

Wenn zum Beispiel ein NFS Server aus irgendwelchen Gründen neu gestartet werden muß, während ein NFS Client eine Datei von einem exportierten Verzeichnis des Servers kopiert, wird der Kopiervorgang des Client nicht abgebrochen, sondern der Client wartet bis der Server wieder antwortet und setzt dann den Kopiervorgang fort.

Dieses Verfahren wird jedoch problematisch, wenn man den Zugriff von mehreren Clients synchronisieren muß, die auf die gleiche Datei schreiben wollen.

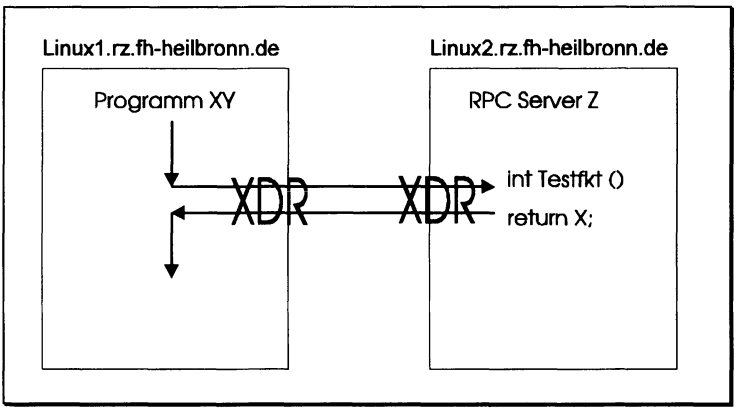
Ein weiterer Nachteil von NFS liegt in der Sicherheit des Protokolls. Neuere verteilte Dateisysteme wie das Andrew Filesystem (AFS), das von OSF DCE verwendet wird, sind NFS hier überlegen. Diese sind jedoch noch nicht so weit verbreitet wie NFS.

Auch unter Linux steht ein Networkfilesystem zur Verfügung. Die entsprechenden Treiber müssen jedoch in den Kernel

hineincompiliert worden sein. Zur Laufzeit müssen dann der `portmap` Daemon sowie der `nfsd` und `mountd` Daemon laufen, um NFS benutzen zu können. Auf welche Verzeichnisse ein NFS-Client zugreifen darf, wird auf dem Server in der Datei `/etc/exports` festgelegt. Näheres dazu findet sich im Kapitel über Konfiguration des Linux-Systems.

3.5 RPC

NFS basiert auf den *Remote Procedure Calls* (RPC) von Sun. Hierbei handelt es sich um einen Mechanismus, mit dem einzelne Routinen auf einem anderen Rechner im Netzwerk ausgeführt werden können. Es gibt dabei einen RPC Server, der Unterprogramme zur Verfügung stellt und Clients, die diese mit Parametern aufrufen. RPC ist also eine spezielle Art der Kommunikation zwischen Prozessen über ein Netzwerk, bei der im Gegensatz zur Netzwerkprogrammierung über Sockets von einem Kommunikationskanal abstrahiert wird.



Ablauf eines RPC Aufrufs

XDR

Zusammen mit RPC wird in der Regel XDR benutzt, eine rechnerunabhängige Art der Datendarstellung, die verwendet wird, um Daten zwischen Rechnern mit unterschiedlicher Prozessorarchitektur auszutauschen. Ein häufiger Unterschied zwischen

zwei verschiedenen Prozessoren ist zum Beispiel die Darstellung von Ganz- oder Integer-Zahlen.
Mit der XDR-Beschreibung können die Parameter einer RPC-Routine definiert werden und dann von entsprechenden Unterprogrammen zwischen dem maschinenunabhängigen und dem eigenen Format eines Rechners konvertiert werden.
Der Aufruf dieser Routinen und der Aufruf der internen RPC-Routinen kann automatisch mit dem Programm `rpcgen` erstellt werden. Es generiert C-Quellcode für den RPC Server und den Client aus einer formalen Beschreibung der RPC-Routinen.
Um eine Verbindung zu einem RPC-Server auf einer anderen Maschine aufbauen zu können, muß auf dieser ein Portmap Daemon laufen. Diesem sind die zur Verfügung stehenden Dienste bekannt, und er leitet die RPC-Aufrufe aus dem Netz an den entsprechenden RPC-Server weiter.

portmap

Die Information, die der Portmap Daemon über registrierte Programme gespeichert hat, kann mit dem `rpcinfo` Programm zu Diagnosezwecken abgefragt werden.

3.6 NIS

Da die konsistente Verwaltung von Accounts (Zugangsberechtigungen) und zugehörigen Passworten innerhalb eines Netzwerkes recht mühsam sein kann, wurde von Sun das sogenannte *Network Information System* (NIS) geschaffen.
Anstatt auf jeder einzelnen Maschine Informationen über die zugangsberechtigten Benutzer, vorhandenen Netzwerkdienste oder andere Systemkonfigurationen zu speichern, werden diese auf einem zentralen NIS-Server verwaltet. Wird nun vom Administrator ein neuer Benutzer eingetragen oder ändert ein Anwender sein Passwort, so sorgt das NIS für die entsprechende Weitergabe dieser Information an alle beteiligten Maschinen.
Ein unter Linux laufender Rechner kann momentan zwar noch nicht als NIS-Server, wohl aber als NIS-Client konfiguriert werden. Auf diese Weise wird die Integration in ein bestehendes UNIX-Netzwerk erheblich vereinfacht.

3.7 Sonstige Netzwerkdienste

ISODE

Wie eingangs bereits erwähnt wurde, existiert für den Aufbau und die Implementierung von Netzwerken ein eigener ISO-Standard. TCP/IP läßt sich zwar in das OSI-Modell einordnen, ist jedoch nicht zu diesem Standard kompatibel. Will man auch unter Linux OSI-konforme Applikationen entwickeln, so kann man auf den frei erhältlichen ISO/OSI-Stack namens ISODE von Marshall T. Rose zurückgreifen, der für Linux portiert wurde.

Novell-Netze,

Um eine Anbindung an vorhandene Novell-Netzwerke (IPX-Protokoll) zur ermöglichen, arbeiten momentan einige Linux-Entwickler an der Implementierung eines entsprechenden Treibers. Erste Alpha-Versionen sind bereits über das Internet verfügbar.

LAN-Manager

Aus einer weiteren Internet-Quelle ist eine Anpassung an den IBM LAN-Manager für Linux verfügbar. Eine Linux-Maschine kann bei dieser Lösimg sowohl als Server als auch als Client fungieren.

Linux Features

Im folgenden Kapitel wird davon ausgegangen, daß der Leser bereits über Grundkenntnisse von UNIX verfügt oder die vorangegangenen Kapitel gelesen hat.
Nun sollen einige wichtige Merkmale und Features von Linux näher beschrieben werden, durch die sich dieses System von anderen UNIX-Varianten aber auch von anderen PC- Betriebssystemen abhebt.

4.1 Virtuelle Konsolen

Viele PC UNIX-Implementationen unterstützen virtuelle Konsolen. Darunter versteht man die Möglichkeit, mehrere voneinander unabhängige Login Shells auf einem Bildschirm verwalten zu können. Die Umschaltung zwischen den einzelnen Sessions erfolgt meist über eine spezielle Tastenkombination.
Unter Linux geschieht dies über die Taste **<Alt>** zusammen mit einer Funktionstaste. Die maximale Anzahl an virtuellen Konsolen ist im Kernel festgelegt. Auf welchen dieser Konsolen ein Login Prompt erscheinen soll, kann in der Datei `/etc/inittab` eingestellt werden.
Unter X-Windows sind die **<Alt>** Tasten für Anwendungen reserviert. Das Umschalten auf eine andere virtuelle Konsole funktioniert jedoch trotzdem mit der Tastenkombination **<Strg-Alt>** und der entsprechenden Funktionstaste. Damit kann zwischen der grafischen Oberfläche und der Textdarstellung der virtuellen Konsolen umgeschaltet werden.

X-Windows

Es ist sogar möglich, auf verschiedenen virtuellen Konsolen, mehrere X-Server zu starten. Dies ist jedoch nicht zu empfehlen, da dafür meist nicht genügend Speicherplatz vorhanden ist und daher die Performance spürbar abnimmt. Statt dessen sollte man unter X11 einen virtuellen Window Manager, wie `olvwm` oder `fvwm`, verwenden, der ebenfalls mehrere virtuelle Bildschirme zur Verfügung stellt.

4.2 Linux-Filesysteme

Die Vielzahl der unter Linux verfügbaren Dateisysteme mag auf den ersten Blick recht verwirrend erscheinen. Die folgenden Abschnitte listen die momentan unterstützten Filesysteme auf und beschreiben deren wichtigste Merkmale.

MINIX-Filesystem

Die ersten Linux-Versionen verfügten nur über einen Filesystemtyp. Dieser war stark an das MINIX-Filesystem angelehnt. So wurde der Aufwand einer kompletten Neuimplementierung umgangen. Außerdem stand auf diese Weise von Anfang an ein stabiles Dateisystem zur Verfügung, das jedoch auch entscheidende Nachteile besitzt.
Dateinamen dürfen nur eine maximale Länge von 14 Zeichen besitzen und die Größe einer Partition ist auf 64 MB begrenzt. Neuere Versionen dieses Linux/MINIX Filesystems erlauben zwar auch längere Dateinamen (30 Zeichen), dennoch dürfte dieses Dateisystem inzwischen kaum noch Verwendung finden.
Bemerkenswert ist allerdings, daß schon diese erste Version eines Linux-Filesystems, im Gegensatz zu vielen kommerziellen System-V-Implementierungen auch Symbolic Links unterstützte.

Extended Filesystem (ext)

Aufgrund der oben erwähnten Einschränkungen, wurde von einem Franzosen das erste alternative Filesystem implementiert. Das sogenannte *Extended Filesystem* (`ext`) unterstützte erstmals

Dateien und Partitionen mit einer maximalen Größe von bis zu 4 GB. Auch die maximale Länge der Dateinamen wurde auf 255 Zeichen erhöht.
Doch auch dieses System hat sein Schwächen. Die Verwaltung der freien Blöcke und i-Nodes erfolgt nicht über einen Bitvektor, sondern über eine verkettete Liste. Dies führt bei längerer Betriebsdauer zu einer übermäßigen Fragmentierung des Speicherplatzes, was sich in einer spürbar längeren Zugriffzeit bemerkbar macht.

Extended2 Filesystem (ext2)

Aus dem Extended ging nach einiger Zeit das *Extended2* Filesystem hervor, das momentan wohl am häufigsten verwendet wird. Die Fragmentierungsprobleme treten in dieser Version nicht mehr auf. Außerdem wird ein Mechanismus unterstützt, der verlorengegangene Sektoren in einem speziellen Unterverzeichnis (`lost+found`) sichert. Eventuelle Systemabstürze und daraus resultierende korrupte Dateisysteme werden beim Starten des Systems erkannt und können über ein spezielles Utility (`e2fsck`) repariert werden.
Aufgrund der flexiblen Struktur des Extended2 Systems werden zukünftige Versionen wahrscheinlich Partitionsgrößen bis zu 4 Terabyte ermöglichen und komprimierte Dateien unterstützen.

Xia Filesystem

Das Extended2 Filesystem blieb nicht der einzige Versuch, ein neues schnelleres Dateisystem zu etablieren. Beinahe zeitgleich tauchte das sogenannte *Xia Filesystem*, benannt nach seinem Autor Frank Xia, auf. Auch dieses erhöht die maximale Partitionsgröße auf 4 GB. Dateinamen können bis zu 248 Zeichen lang sein. Die Größe einer Datei ist jedoch momentan auf 64 MB beschränkt.

Andere Filesysteme

DOS & OS/2

Das DOS-Filesystem ermöglicht den transparenten Zugriff auf DOS-Disketten oder Partitionen (auch OS/2-FAT-Partitionen) . Auf diese Weise kann auf vorhandene Datenbestände zurückgegriffen werden, wie dies in einem der folgenden Abschnitte näher erläutert wird.

System V

Für den Zugriff auf Partitionen von Xenix- bzw. System V und OS/2-HPFS wurden ebenfalls spezielle Dateisysteme entwickelt, die jedoch zum Teil noch nicht vollständig implementiert sind.

ISO 9660/HighSierra Filesystem

Um den Zugriff auf CD-ROMs zu ermöglichen, stellt Linux sowohl ein ISO9660-kompatibles als auch das High Sierra Filesystem zur Verfügung. Auch die sogenannten *Rockridge Extensions* zur Unterstützung längerer Dateinamen wurden implementiert.

Proc-Filesystem

Ein besonderes Dateisystem, das nicht der Verwaltung von Dateien dient, sondern den Zugriff auf Informationen des Kernels und der momentan erzeugten Prozesse ermöglicht, ist das Proc-Filesystem. Es wird meistens beim Hochfahren des Systems auf das Verzeichnis `/proc` im Wurzelverzeichnis abgebildet.
Dieses enthält für jeden laufenden Prozeß wiederum ein Unterverzeichnis mit dem Namen der entsprechenden Prozeß-Id. Die darin befindlichen Dateien stellen eine flexible Schnittstelle zu den eigentlichen Prozeß-spezifischen Informationen dar. Im allgemeinen handelt es sich dabei um virtuelle ASCII-Dateien. Der Inhalt läßt sich beispielsweise über das `cat`-Kommando ausgegeben. Auf diese Weise kann man den Inhalt der Kommandozeile oder die für einen Prozeß gültigen Environment Variablen ermitteln. Auch Informationen über den Speicherplatzbedarf, den Vaterprozeß oder den aktuellen Prozeßzustand lassen sich so gewinnen.

4.3 Datenaustausch

In den seltensten Fällen wird Linux als einziges Betriebssystem auf einem PC verwendet. Häufig ist auf einer anderen Partition oder Festplatte noch DOS mit MS-Windows, OS/2 oder ein anderes PC-UNIX installiert. Wer von DOS auf Linux umsteigt, möchte häufig nicht auf seine alten Programme verzichten.
Die folgenden Abschnitte sollen zeigen, wie man neben Linux mit verschiedenen Betriebssystemen arbeiten kann und wie Daten und Programme zwischen diesen Betriebssystemen ausgetauscht werden können.
Eines der wichtigsten Utilities bei der Verwendung mehrerer Betriebssysteme auf einem Rechner ist ein Bootmanager. Er ermöglicht es, beim Starten des Systems das zu bootende Betriebssystem auszuwählen. Bei Linux erfüllt der Linux Loader (LILO) unter anderem diese Funktion. Seine Bedienung und Installation wird im Kapitel *Installation* genauer beschrieben.

Bootmanager

MTools

Da gerade der Austausch von Dateien mit DOS-Systemen eine Anforderung ist, die heute an fast alle Betriebssysteme gestellt wird, gibt es schon seit einiger Zeit frei verfügbare Programme zur Verarbeitung von DOS-Dateien unter UNIX.
Wie auf vielen anderen UNIX-Systemen existieren zu diesem Zweck auch unter Linux die sogenannten MTools. Dabei handelt es sich um Befehle wie `mdir` oder `mcopy`, mit denen das Verzeichnis eines DOS-Datenträgers, typischerweise einer Diskette, gelesen bzw. Dateien kopiert werden können.

```
dirk1:/home/stefan# mdir a:
 Volume in drive A is dosdisk1
 Directory for A:/

COMMAND  COM       55591    3-10-93   6:00a
WINA20   386        9349    6-11-91  12:00p
AUTOEXEC BAT         359    8-26-93   9:02p
CONFIG   SYS         377    5-23-93   2:48p
DOSKEY   COM        6012    6-11-91  12:00p
EDIT     COM         429    6-11-91  12:00p
FORMAT   COM       34223    6-11-91  12:00p
       9 File(s)     1270784 bytes free
dirk1:/home/stefan# mcopy -t a:autoexec.bat .
Copying AUTOEXEC.BAT
dirk1:/home/stefan# mdel a:autoexec.bat
dirk1:/home/stefan#
```

Zugriff auf eine MS-DOS-Diskette mit den MTools

DOS-Filesystem

Linux stellt neben den MTools eine weitere Methode zum Zugriff auf DOS Datenträger zur Verfügung, das bereits erwähnte DOS-Filesystem. Damit ist es möglich, Disketten und DOS-Partitionen einer Festplatte wie andere Dateisysteme in den Linux Verzeichnisbaum einzuhängen (mounten). Man erreicht also einen völlig transparenten Zugriff auf die darauf enthaltenen Daten.
Hinzu kommt eine höhere Zugriffsgeschwindigkeit im Vergleich zu den MTools, da die Ein- und Ausgabeoperationen jetzt vom Cache des Betriebssystems profitieren können.

```
dirk1:/# cd msdos
dirk1:/msdos# ls -a
./   ../
dirk1:/msdos# cd ..
dirk1:/# mount -t msdos /dev/hda2 /msdos
dirk1:/# cd msdos
dirk1:/msdos# ls -a
./             command.com*   format.com*    tools/
../            config.sys*    io.sys*        wina20.386*
autoexec.bat*  dos/           msdos.sys*     windows/
dirk1:/msdos# cd dos
dirk1:/msdos/dos#
```

Zugriff auf eine MS-DOS-Partition mit dem DOS-Filesystem

Der Nachteil beim Mounten von Disketten ist jedoch, daß diese nicht mehr nach Belieben eingelegt und entfernt werden können, sondern jeweils die Befehle `mount` und `umount` verwendet werden müssen. Wird eine Diskette, während sie gemountet ist,

gewechselt, so wird dadurch meist die danach eingelegte Diskette überschrieben.

Zugriffsrechte auf das DOS Filesystem

Da es unter DOS keine Benutzer- oder Gruppen-Id gibt, können den einzelnen Dateien eines gemounteten DOS-Filesystems keine individuellen Einstellungen für die Zugriffskontrolle gegeben werden. Statt dessen lassen sich beim Mounten die Group und User Id, sowie die Zugriffsrechte für das gesamte Filesystem als Option angeben. Damit kann zumindest der Zugriff auf das Filesystem als ganzes geregelt werden.

Textkonvertierung

Zeilenumbruch (CR/LF)

Ein grundsätzliches Problem beim Datenaustausch zwischen DOS und UNIX ist die unterschiedliche Darstellung des Zeilenumbruchs bei Textdateien. Bei `mcopy` wird dies durch die Kommandozeilenoption `-t` gelöst, die angibt, ob eine Datei im Binärmodus oder im Textmodus mit Konvertierung kopiert werden soll.
Da im Falle eines gemounteten DOS-Filesystems auf sehr viele verschiedene Dateien zugegriffen werden kann, gibt es hier keine universelle Lösung. Es gibt zwar Optionen für den `mount`-Befehl, mit denen eine automatische Konvertierung aktiviert werden kann, doch funktioniert dieses Verfahren nicht immer zuverlässig. Es ist nämlich nicht immer sicher entscheidbar, ob es sich bei einer Datei um eine Text- oder eine Binärdatei handelt.

4.4 Emulatoren

Der Austausch von Daten zwischen Linux und anderen Betriebssystemen alleine reicht vielen Anwendern nicht aus. Sie benötigen zusätzlich Zugriff auf Programme, die für MS-DOS, MS-Windows oder andere UNIX-Systeme geschrieben wurden. Die Unterstützung, die Linux in diesem Bereich bietet, wird im folgenden beschrieben.

DOS-Emulator

Wie bei OS/2 und anderen neueren Betriebssystemen existiert für Linux ein DOS-Emulator, also die Möglichkeit, DOS-Programme gleichzeitig mit anderen Linux-Anwendungen ablaufen zu lassen. Dieser bildet jedoch nicht das DOS-Betriebssystem nach, sondern nur die rudimentären Ein-/Ausgaberoutinen (BIOS) und ermöglicht den Zugiff auf alle wichtigen Geräte.

Mit dieser Systemerweiterung kann unter der Kontrolle von Linux DOS gebootet werden. Die DOS-Version ist dabei eher von untergeordneter Bedeutung.

Grafik, HMA, UMB und EMS

Obwohl der Emulator noch weit davon entfernt ist, jedes DOS Programm problemlos zu unterstützen, sind die Möglichkeiten schon sehr beachtlich. So kann man inzwischen Programme im Grafikmodus und mit Unterstützung von High Memory, Upper Memory Blocks (UMB) und EMS ablaufen lassen. Programme wie der Norton Commander und der Texteditor QEdit arbeiten problemlos, aber auch größere kommerzielle Produkte wie Turbo Pascal oder Word Perfect können verwendet werden.

Da die Umschaltung der virtuellen Konsolen auch innerhalb des Emulators funktioniert, kann unter Linux zwischen mehreren DOS- und UNIX-Applikationen umgeschaltet werden.

```
 1 MB display memory

Linux DOS emulator 0.49 $Date: 1993/05/04 05:29:22 $
Last configured at Sat Jul 31 18:40:22 1993
                 on softland, Linux 0.99.11 #3 Sat Jul 24
08:48:18 GMT 1993
maintained by Robert Sanders, gt8134b@prism.gatech.edu

[Linux File System] drive C: id directory /msdos/
Welcome to the Linux DOS emulator version 0.49!
A:\>
A:\>mem


    655360 Byte konventioneller Speicher insgesamt
    655360 Byte für MS-DOS verfügbar
    638448 Byte max. Größe für ausführbares Programm

   1024000 Byte fortlaufender Erweiterungsspeicher insgesamt
         0 Byte fortlaufender Erweiterungsspeicher verfügbar
   1024000 Byte XMS-Speicher verfügbar
           MS-DOS resident im oberen Speicherbereich (High
Memory Area)
```

Ausgabe des DOS-Emulators

Neben normalen Disketten und DOS-Partitionen kann vom DOS-Emulator auch auf sogenannte Disk-Images und Festplatten-Images zugegriffen werden. Für DOS-Programme verhalten sich diese Dateien wie "echte" Datenträger. So kann zum Beispiel über das Disk-Image einer Bootdiskette das DOS-System gestartet werden und gleichzeitig das physikalische Laufwerk A als virtuelles Laufwerk B verwendet werden. Dadurch benötigt man keine Diskette um den Emulator zu starten. Im Gegensatz zum Booten von einem Festplatten-Image kann dann auch die eigentliche DOS-Partition als Laufwerk C angesprochen werden.

Außer dem Zugriff auf echte DOS-Datenträger und Imagedateien kann auch mit einem speziellen Gerätetreiber auf die Linux-Filesysteme zugegriffen werden. Dieser Treiber, der meist als `emufs.sys` auf einem Festplatten-Imagefile vorhanden ist, arbeitet ähnlich wie ein Redirector beim Zugriff auf Netzwerklaufwerke. Das Linux-Filesystem, oder ein bestimmtes Unterverzeichnis erscheint dabei unter dem nächsten freien Laufwerksbuchstaben.

emufs.sys

Dadurch wäre es möglich, auf eine DOS-Partition einer Festplatte sowohl direkt über den DOS-Emulator, als auch über das Linux-Filesystem zuzugreifen, sofern das DOS-Filesystem gemountet ist. **Vorsicht!** Dies sollte jedoch auf jeden Fall unterlassen werden, um Datenverluste zu vermeiden. Falls man sowohl von Linux als auch vom Emulator aus auf eine DOS-Partition zugreifen möchte, so sollte der Emulator über den Gerätetreiber und das Linux-Filesystem auf die DOS-Partition zugreifen und nicht direkt.

Die Konfiguration des DOS-Emulators erfolgt größtenteils durch Editieren der Datei `/etc/dosemu/Config`, die beim Starten des Emulators gelesen wird. Darin kann zum Beispiel eingestellt werden, von welchem Laufwerk oder aus welcher Datei der Emulator booten soll und welche I/O-Ports angesprochen werden dürfen.

Konfiguration

X11-DOS-Emulator

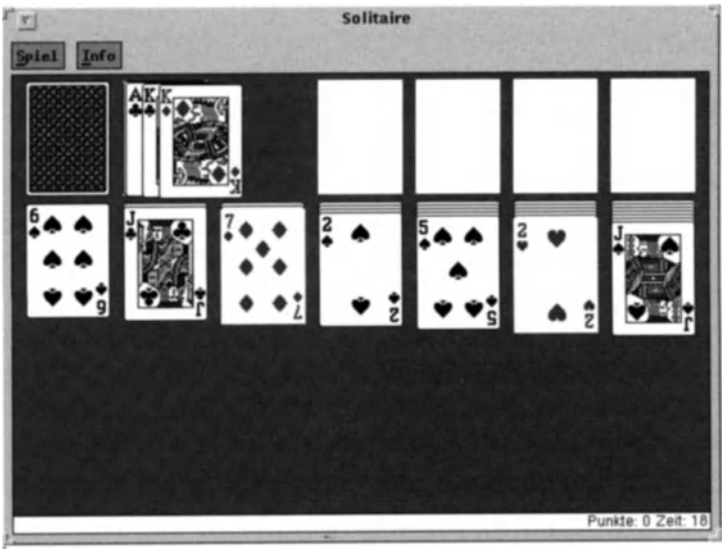
Neben dem eigentlichen DOS-Emulator, der auf einer Konsole im Text- bzw. Grafikmodus abläuft, existiert eine weitere Varian-

te, die es erlaubt, ein DOS-Programm in einem X-Windows-Fenster ablaufen zu lassen. Damit ist es auch möglich, über die Umleitung der X-Windows-Ausgaben von einem anderen Rechner im Netzwerk, DOS-Programme zu benutzen, die unter dem DOS- Emulator laufen.
Da zur Darstellung der Bildschirmausgaben eine Variante des Color-XTerm-Programms benutzt wird, verfügt der X11-DOS-Emulator auch über eine scrollbare History. Außerdem lassen sich mit der Maus Bildschirmausgaben ausschneiden und in ein anderes Fenster kopieren.
Leider unterstützt die X11-Variante keine DOS-Programme, die im Grafikmodus laufen. In diesem Fall muß auf die normale Version des Emulators zurückgegriffen werden, die auch eine höhere Performance besitzt.

WINE

Eine Entwicklung, die wohl großen Einfluß auf die Verbreitung von UNIX auf PCs sowohl im privaten, als auch im kommerziellen Bereich ausüben wird, ist das sogenannte Windows Application Binary Interface (WABI).
Dieses Interface wurde von Sun entwickelt und von den Unix System Laboratories (USL) lizensiert, sodaß es auch für System V UNIX verfügbar sein wird. Es bietet die Möglichkeit, MS-Windows-Programme direkt unter UNIX ablaufen zu lassen. Dazu werden die Windows-API-Aufrufe in entsprechende X-Windows-Calls übersetzt, was den interessanten Nebeneffekt hat, daß diese Programme wegen der Netzwerktransparenz von X11 auch von anderen X-Servern aus benutzt werden können.
Bereits Anfang Juni 1993 kamen in der Mailing Liste des Linux DOS-Emulators Diskussionen über eine Portierung oder Eigenentwicklung eines WABI für Linux namens WINE auf. Nach wenigen Tagen wurde für dieses Projekt eine eigene Mailing-Liste eingerichtet und mit der konkreten Entwicklung begonnen.
Einige Wochen später war ein Windows Program Loader weitgehend vollendet und man konnte erste Windows-Testprogramme, die ein Fenster mit einem Menü darstellen, unter Linux ablaufen

lassen. Mittlerweile ist auch schon das erste richtige Windows-Programm (Solitär) unter Linux lauffähig.



MS-Windows Solitär unter WINE

ELF Support

Neben einer Emulation der Xenix-Systemaufrufe, die schon für sehr frühe Linux-Versionen verfügbar war, gibt es eine Emulationsbibliothek für System V Binaries, die das ELF-Format verwenden. ELF ist das neue Linkformat für System V Release 4 Programme und Bibliotheken. Die Entwicklung dieser Library ist jedoch noch nicht abgeschlossen.
Auch an der Unterstützung von Programmen im COFF Format, wie es die meisten kommerziellen UNIX-Versionen auf PCs benutzen, wird momentan gearbeitet.

4.5 Alternative Shells

Wie erwähnt, werden unter Linux selten die originalen UNIX-Shells `sh` bzw. `csh` benutzt. Dies liegt zum einen daran, daß die

Quelltexte nicht frei erhältlich sind und zum anderen sind mittlerweile erheblich komfortablere Versionen entwickelt worden.

Bash

Die `bash` (Bourne Again Shell) entstammt, wie viele andere Programme, dem GNU-Projekt der Free Software Foundation und kann als Erweiterung der Korn-Shell betrachtet werden, die auch viele Features der C-Shell kennt. Darüberhinaus sind praktische Spezialfunktionen wie automatische Namens- oder Pfaderweiterung oder Historyscrolling über die Pfeiltasten möglich. Dies soll an einem Beispiel näher erläutert werden.

automatische Pfaderweiterung

Um vom Verzeichnis `/home/stefan` in das Verzeichnis `/usr/src/linux` zu wechseln, kann der Befehl `cd /usr/src/linux` natürlich manuell, Buchstabe für Buchstabe eingegeben werden. Mit der automatischen Pfaderweiterung der `bash` reicht es jedoch, Verzeichnisse nur so weit einzugeben, bis sie eindeutig zugeordnet werden können. Dann kann durch Eingabe von **<Tab>** der Pfad erweitert werden.

```
/home/stefan> cd /usr/s
```

wird nach Eingabe von **<Tab>** zu

```
/home/stefan> cd /usr/src/
/home/stefan> cd /usr/src/l
```

wird nach Eingabe von **<Tab>** zu

```
/home/stefan> cd /usr/src/linux/
```

um in diesem Verzeichnis das Makefile des Linux-Kernels zu editieren, reicht ebenfalls die Eingabe von wenigen Zeichen.

```
/usr/src/linux> emacs M
```

wird nach Eingabe von **<Tab>** zu

```
/usr/src/linux> emacs Makefile
```

da das Makefile in diesem Verzeichnis die einzige Datei ist, die mit einem großen M beginnt.

Falls es bei der Erweiterung mehrere Alternativen gibt, ertönt ein Warnton und die bash zeigt nach nochmaligem Drücken der **<Tab>** Taste alle möglichen Varianten an.

Mit den Pfeiltasten (auf und ab) kann in der Kommandohistorie geblättert werden.
So erscheint am obigen Beispiel nach einmaligem Drücken der Pfeil-hoch-Taste

Kommandohistorie

```
/usr/src/linux> emacs Makefile
```

und nach nochmaligem Drücken

```
/usr/src/linux> cd /usr/src/linux/
```

Die Anzeige des Hostnamen und des aktuellen Pfades als Prompt ist durch die Verwendung von Environment Variablen möglich. Dazu fügt man zum Beispiel die folgende Zeile in die Datei .bashrc im jeweiligen Home-Verzeichnis ein.

```
PS1='\h:$PWD>'
```

Die Bourne Again Shell bietet noch viele weitere Features, die hier jedoch nicht ausführlich beschrieben werden können. Für Details sei auf die Manual page der bash verwiesen.

TCSH

Eine Alternative zur Bourne Again Shell ist die tcsh. Sie ist eine Erweiterung der C-Shell und kann wie die bash mit der **<TAB>** Taste Pfade und Befehle erweitern und mit den Pfeiltasten in der Kommandohistorie scrollen oder die Zeilen

editieren. Außerdem lassen sich verschiedene Zusatzfunktionen aktivieren, wie beispielsweise ein Watch Modus, in dem eine Meldung ausgegeben wird, falls sich ein Benutzer ein- oder ausloggt. Die Ausgabe aller Alternativen bei der automatischen Erweiterung wird bei der tcsh mit **<Strg-D>** aufgerufen, sämtliche Tasten sind jedoch konfigurierbar. So kann man zum Beispiel mit

```
linux1:/home/tul> cd /usr/src/ <Strg D>
```

das Inhaltsverzeichnis des /usr/src Verzeichnisses ausgeben, bevor man den Befehl cd ganz eingegeben hat.
Ein weiteres interessantes Feature ist die Ausgabe der Manual pages zu Befehlen, die man gerade verwenden möchte.
Ungewöhnlich ist auch die Möglichkeit der automatischen Korrektur von falsch eingegeben Kommandos oder Dateinamen. Die tcsh versucht dabei, den nächsten passenden Befehl oder Dateinamen zu benutzen.

4.6 Erweiterte Kommandos

Viele der Standard-UNIX-Befehle, die bei Linux verwendet werden entstammen dem GNU-Projekt und sind Erweiterungen der normalen Befehle. Der Befehl ls zum Beispiel hat unter Linux über 20 Optionen, mit denen bei Bedarf die Art der Anzeige, Sortierung oder Behandlung von symbolischen Verweisen geändert werden kann.

```
LS(1L)                                                    LS(1L)

NAME
       ls, dir, vdir, ll, lsf - list contents of directories

SYNOPSIS
       ls  [-abcdgiklmnpqrstuxABCFLNQRSUX1]  [-w  cols] [-T cols]
       [-I pattern] [--all]  [--escape]  [--directory]  [--inode]
       [--kilobytes]  [--numeric-uid-gid]  [--hide-control-chars]
       [--reverse]   [--size]   [--width=cols]   [--tabsize=cols]
       [--almost-all]  [--ignore-backups]  [--classify]  [--file-
       type]   [--ignore=pattern]   [--dereference]   [--literal]
       [--quote-name]                                [--recursive]
       [--sort={none,time,size,extension}]                 [--for-
       mat={long,verbose,commas,across,vertical,single-column}]
       [--time={atime,access,use,ctime,status}] [path...]
...
```

Ausschnitt aus der Manual page des GNU ls-Befehls

GNU Tar

Der `tar`-Befehl kennt in der GNU-Variante beispielsweise eine Option `z`, mit der tar-Archive automatisch komprimiert und dekomprimiert werden können oder eine Option `M`, mit der ein Archiv über mehrere Datenträger verteilt werden kann. (Multi Volume Archiv)

Gzip

Der Befehl `compress` kann unter Linux durch `gzip` ersetzt werden, einem Programm das kompatibel zu mehreren anderen Komprimierungsverfahren ist und eine deutlich höhere Effektivität als `compress` besitzt. Ein komprimiertes tar-Archiv (Das Kommando `tar` ermöglicht die Zusammenfassung einzelner Dateien und Unterverzeichnisse zu einem einzigen Archiv), das mit dem normalen UNIX-Befehl `compress` eine Größe von 1,5 MB besaß, benötigt bei Komprimierung mit `gzip` häufig nur 900 KB.

Alle diese erweiterten Befehle sind natürlich im Quellcode verfügbar und lassen sich auch auf anderen UNIX Maschinen übersetzen. Der Vorteil bei Linux ist jedoch, daß diese von Anfang an verwendet werden und kein weiterer Aufwand in die Konfiguration, Compilation und Installation dieser Utilities investiert werden muß.
Die Liste aller erweiterten Befehle mit ihren Optionen würde sicher ein eigenes Buch füllen. Daher soll hier auf die Manual pages zu den einzelnen Befehlen verwiesen werden.

Installation

In diesem Kapitel werden die wichtigsten Bezugsquellen für Linux sowie die Installation und im nächsten Kapitel die Konfiguration des Systems beschrieben. Dabei wird nicht auf Details der Installation spezieller Linux-Installationspakete eingegangen, sondern ein grundlegender Überblick vermittelt, der für die verschiedenen Pakete (Distributionen) gleichermaßen gültig ist.

5.1 Pakete

Da Linux als Free Software aus vielen Teilen besteht, die von verschiedenen Leuten in der ganzen Welt unabhängig voneinander entwickelt werden, gibt es keine "offizielle" Linux-Distribution, in der alle für Linux verfügbaren Programme zusammengefaßt sind.
Statt dessen gibt es eine Reihe von FTP-Servern, von denen der Linux Kernel, verschiedene Utilities und Programme, aber auch komplette Installationspakete über das Internet bezogen werden können.
Fast alle diese Installationspakete oder Releases umfassen zumindest den Kernel und alle wichtigen Utilities und Programme, die man für eine Erstinstallation benötigt. Darüber hinaus enthalten sie oft den GNU C-Compiler, die grafische Oberfläche, sowie viele weitere Programme.

SLS

Das wohl bekannteste und der erste de facto Standard ist das SLS (Softlanding Systems) Release. Es umfaßt derzeit ca. 30 Disketten, die installiert zwischen ca. 10 und 90 MB Plattenplatz benötigen, je nachdem, welche Komponenten installiert werden. Enthalten sind, neben den grundlegenden Befehlen und Utilities, das X11-System, die Entwicklungsumgebung, der Quellcode des Kernels sowie viele zusätzliche Programme.

Installation auch über NFS

Die Installation erfolgt mit einem einfach zu bedienenden Programm und kann entweder direkt von den entsprechenden Medien, also Diskette, Tape oder CD, oder per NFS von einem anderen Rechner erfolgen. Mit diesem Programm werden bereits bei der Installation die wichtigsten Konfigurationsdateien interaktiv angepaßt. Der Zeitaufwand für eine solche Installation liegt je nach Erfahrung zwischen einer Stunde und mehreren Stunden.

Die jeweils aktuellste Version ist in dem Verzeichnis `/pub/linux/packages/SLS` immer auf dem FTP-Server `tsx-11.mit.edu` verfügbar. Außerdem kann sie als Diskettenpaket, Tape oder CD bei SLS bestellt werden.

Das SLS-Paket wird auch immer häufiger vom deutschen Versandhandel und im Anzeigenteil von Fachzeitschriften angeboten.

MCC Interim

Ein anderes Installationspaket ist das MCC Interim Release. Es stammt von der Universität in Manchester und zeichnet sich gegenüber der SLS-Distribution dadurch aus, daß viele der weniger wichtigen Programme weggelassen wurden. In diesem Paket ist auch kein X11 und kein TeX enthalten.

Damit eignet es sich besonders für Anwender, die weitere Programme gerne selbst installieren und damit genau wissen, was auf ihrer Festplatte installiert ist. Für Anfänger ist es dagegen weniger geeignet.

Slackware

Eine Distribution neueren Datums ist das Slackware-Paket. Anfänglich basierte es sehr stark auf dem SLS Release. Mittlerweile werden aber eigene Installationsroutinen benutzt, die einen sehr ausgereiften Eindruck machen. So kann der Benutzer beispielsweise gleich zu Beginn der Installation eine landesspezifische Tastaturtabelle auswählen. Auch die Slackware-Distribution erlaubt eine Installation von Disketten, Festplatte und über NFS.
Der Umfang des Paketes ist nochmals größer als der des SLS. Neben dem Basissystem wird auch eine Diskettenserie mit Spielen und dem neusten GNU-Emacs angeboten. Der Verwalter des Slackware-Paketes achtet sorgfältig darauf, daß jeweils die aktuellsten Programmversionen enthalten sind, was jedoch bei der schnellen Entwicklung des Linux-Systems nicht immer leicht sein dürfte.

Linux/CV

Eine relativ neue Distribution aus Deutschland ist Linux/CV. Diese Linux-Variante der *Free Software Association Germany* (FSAG) wendet sich vor allem an kommerzielle Interessenten. Ziel ist es, eine besonders stabile Installation zu ermöglichen, die nicht unbedingt den neusten Kernel enthalten sollte. Durch das zeitweilige Einfrieren der Kernelversion soll auch die Entwickung bzw. Portierung kommerzieller Software auf Linux erleichtert werden. Jedem Linux/CV-Paket liegen die OSF/Motif-Bibliotheken bei, so daß diese Distribution nicht kostenlos erhältlich sein kann.
Zentraler Bestandteil von Linux/CV ist G.R.E.A.T. Dabei handelt es sich um einen grafischen Desktop, der auch dem unerfahrenen Anwender den problemlosen Umgang mit Linux ermöglichen soll.

G.R.E.A.T

CDs

Viele Pakete werden inzwischen auf CD bzw. als CD-Abonnement angeboten. Dabei erhält man mehrmals im Jahr eine neue CD mit der aktuellen Version des Linux Kernels und einiger Programme oder auch ganze Kopien des Linux-Verzeichnisbaums eines FTP-Servers.
Ein bekanntes Beispiel ist die CD von Jana Publishing oder die CD von Yggdrasil Computing, auf der neben einem bootfähigen und installierten Linux-System der Quellcode des MIT X11R5 Systems und viele GNU-Utilities enthalten sind.
Die Handhabung einer CD ist natürlich einfacher als die eines Paketes mit 30 Disketten. Man sollte jedoch bedenken, daß Linux sich von Tag zu Tag weiterentwickelt, und daher in den Wochen oder sogar Monaten, die vom Erstellen der CD bis zum Versand vergehen, schon wieder eine neue Version mit vielen Verbesserungen vorhanden sein kann.

Unifix

Eine besonders interessante CD ist die Distribution der Firma Unifix. Der Käufer erhält neben der CD und einem kleinen Handbuch eine Bootdiskette, über die der erste Systemstart erfolgt. Nach der Installation einer etwa 5 MB großen Minimalkonfiguration auf einer Partition der Festplatte ist das System einsatzbereit. Sämtliche Anwendungsprogramme können direkt von der CD geladen werden. Verfügt der jeweilige Rechner über genügend Hauptspeicher (8-16 MB), so werden die wichtigsten Daten der CD im Speicher gehalten, was sich äußerst positiv auf die Zugriffgeschwindigkeit auswirkt. Bei eventuellen Updates genügt dann der Wechsel der CD. Es entfällt die zeitraubende Installation der kompletten Linux-Software.

5.2 Bezugsquellen

Da der Bekanntheitsgrad von Linux in letzter Zeit stark gestiegen ist, gibt es inzwischen auch viele Möglichkeiten, die oben genannten Pakete uns zusätzliche Programme für Linux zu beziehen. Der direkteste und schnellste Weg ist jedoch mit Sicherheit der Zugriff auf einen FTP-Server im Internet.

FTP-Server

Der wichtigste FTP-Server für Linux in Europa ist `nic.funet.fi` in Finnland, der der erste Linux-Server überhaupt war. Auf ihm finden sich die neusten Kernel Versionen und eventuelle neuere ALPHA-Releases. Der `tsx-11.mit.edu` enthält den jeweils aktuellen GNU C-Compiler und die Linux-C- Libraries. Ein breites Angebot an Linux-Software bietet auch `sunsite.unc.edu` in den USA. Dort werden vor allem viele nach Linux portierte Softwarepakete in das `Incoming-` Verzeichnis geladen.

Diese Server werden auch von mehreren deutschen FTP-Servern gespiegelt. Das bedeutet, daß die Dateien der FTP-Server in den USA in regelmäßigen Abständen auf den entsprechenden deutschen FTP-Server kopiert werden. Damit soll vermieden werden, daß das Internet unnötig belastet wird. Meist ist auch eine innereuropäische Verbindung schneller und sicherer als der direkte Zugriff auf die amerikanischen Server.
Derartige Spiegel-Server in Deutschland sind beispielsweise:

- `fgb1.fgb.mw.tu-muenchen.de`
- `ftp.dfv-rwth.aachen.de`
- `ftp.informatik.hu-berlin.de`

Einen aktuellen Überblick über Linux-FTP-Server in Deutschland erhält man aus der Linux FAQ oder über die deutsche Linux-Newsgruppe `de.comp.os.linux`.

FTP Mailserver

Wer keinen direkten Internet-Zugang zur Verfügung hat, jedoch E-Mail senden und empfangen kann, hat die Möglichkeit, über sogenannte FTP-Mailserver auf die Internet-FTP-Server zuzugreifen.
Ein FTP-Mailserver verschickt Programme von FTP-Servern per E-Mail. Dazu sendet man einen entsprechenden Befehl an die E-Mail-Adresse des Servers, der dann auf den FTP-Server zugreift, das Programm in kleinere Stücke zerlegt, und diese meist mit `uuencode` kodiert als E-Mail zurückschickt.

Die Befehle, die ein solcher Server versteht, bekommt man in der Regel durch Senden einer Nachricht mit dem Kommando `help` in der ersten Zeile.
Derartige Mailserver sind zum Beispiel unter den Adressen `ftp-mailer@informatik.tu-muenchen.de` oder aber `ftpmail@decwrl.dec.com` erreichbar.
Zu beachten ist jedoch, daß sich ein FTP-Mailserver nicht dazu eignet, ein komplettes Linux SLS Release zu übertragen. Diese Server haben in der Regel eine Begrenzung auf kleine Datenmengen.

Versandhandel

Andere Bezugsquellen für Linux und Linux-Programme sind verschiedene kommerzielle Händler, die Linux-Disketten oder CDs im Versand anbieten. Die Adressen solcher Händler findet man am einfachsten in den deutschen Fachzeitschriften.
Teilweise werden von diesen Händlern sogar Festplatten angeboten, auf denen bereits ein lauffähiges Linux-System installiert ist.

Mailboxen

Inzwischen gibt es eine Reihe von Mailboxen in größeren Städten, die sich auf Linux spezialisiert haben oder zumindest einige Linux-Programme anbieten. Eine spezielle Liste (BBS List) enthält eine Übersicht über viele Mailboxen in aller Welt. Sie findet sich beispielsweise in den Linux Installationsdokumenten .

5.3 Hardware

Eine der am häufigsten gestellte Fragen ist, welche Hardware von Linux unterstützt bzw. benötigt wird. Die Grundvorraussetzung ist ein PC–kompatibler Rechner mit einem 386er oder neueren Prozessor. Auf alten XTs oder ATs mit 286er Prozessor läuft Linux definitiv nicht.

Hauptspeicher

Die Minimalausstattung ist ein 386SX Rechner mit 2 MB Speicher. Dabei wird die Installation jedoch unter Umständen schwierig, da 2 MB Speicher für viele Programme nicht ausreichen. In diesem Fall muß möglichst früh eine Swap Partition oder ein Swap File eingerichtet werden, um überhaupt die für die Konfiguration nötigen Programme und Editoren starten zu können.

Ein normales Arbeiten im Textmodus ist ab 4 MB Hauptspeicher möglich, und um mit X-Windows in der normalen Version arbeiten zu können, sollten schon 8 MB zur Verfügung stehen. Ist der Rechner sogar mit 16 MB ausgerüstet, so macht sich dies durch einen deutlichen Performancegewinn vor allem unter X11 bemerkbar.

8 MB Speicher oder mehr

Festplatte

Was den benötigten Platz auf der Festplatte angeht, so kann man für eine minimale Installation, unter Verzicht auf die grafische Oberfläche, mit 40 MB auskommen.

Um jedoch eine Komplettinstallation des SLS-Pakets mit X11, dem C-Compiler und allen Tools und Utilities durchzuführen, sollte man ca. 100 MB für Linux verfügbar haben. Nach oben sind hier keine Grenzen gesetzt. Wer Zugang zum Internet hat, wird kein Problem haben, eine 500 MB Festplatte mit Free Software zu füllen. Das Angebot an Programmiersprachen, Utilities, Bibliotheken und Anwendungsprogrammen ist mittlerweile enorm groß.

100 MB auf der Festplatte

Für eine preiswerte Erstinstallation ist es empfehlenswert, auf eine Festplatte mit AT-Bus-Interface zurückzugreifen. Diese sind in vielen Versionen bis ca. 500 MB erhältlich und bieten eine ausreichende Performance.

Dieses Interface wurde von Anfang an vom Kernel unterstützt, und benötigt keinen weiteren Treiber. Es existiert sogar ein Kernel-Patch, mit dem es möglich ist, auch zwei AT-Bus

Festplatten mit AT-Bus Interface

Controller parallel zu betreiben, sofern deren I/O-Ports und Interrupts konfigurierbar sind.

SCSI

Da die Übertragungsrate von AT-Bus Festplatten begrenzt ist, und sie aufgrund ihrer Architektur eine maximale Kapazität von ca. 500 MB bieten, wird für größere und schnellere Platten in der Regel das *Small Computer System Interface* (SCSI) verwendet. Es erlaubt die parallele Nutzung von bis zu sieben Geräten. Dabei spielt es keine Rolle, ob es sich um Festplatten, Bandlaufwerke oder einen Scanner handelt. Zum Betrieb wird jedoch ein spezieller Treiber für den SCSI Hostadapter im Linux Kernel benötigt, der aber mittlerweile in allen aktuellen Versionen bereits enthalten ist.

übliche SCSI Hostadapter

Man sollte daher keine exotischen Hostadapter verwenden, da man dafür nur selten einen Treiber findet. Eventuell mitgelieferte Treiber für MS-DOS oder andere UNIX-Versionen sind nicht verwendbar.
Ohne Probleme arbeiten die Controller 1542b und 1742 (EISA) der Firma Adaptec. Außerdem existieren Treiber für Seagate ST02, Future Domain und andere. Hier sollte man im Zweifelsfall die jeweils aktuelle Liste der unterstützten Hardware zu Rate ziehen.

Grafikkarten

VGA-Karte mit ET4000 Chipsatz

Bei der Grafikkarte ist die unkomplizierteste Lösung, auf eine einfache Karte mit einem ET4000 Chipsatz von Tseng zurückzugreifen. Die Geschwindigkeit ist für normale Anwendungen ausreichend, da der X-Server spezielle Optimierungen für diesen Chipsatz enthält. Kompatibilitätsprobleme sind bei diesen Karten bisher nicht bekannt. Andere direkt unterstützte Chipsätze sind ET3000, PVGA, Trident und GVGA. Ab XFree 2.0 können auch Grafikkarten der Firma Cirrus Logic eingesetzt werden. Der generische VGA-X-Server für 16 Farben sollte eigentlich mit jeder VGA-Karte funktionieren.

Eine wesentlich höhere Performance bieten Karten mit S3 oder Mach8/32 Chipsatz. Besonders schnell sind diese Karten in der Local Bus Variante, wie die ATI Ultra Pro oder die ELSA Winner. Es lassen sich mit diesen neuen Grafikkarten Benchmarkwerte von 100 bis 150 000 XStones erreichen.
Problematisch sind eventuell Karten des Herstellers Diamond, da dieser Hersteller die nötigen Spezifikationen seiner Karten nicht frei verfügbar macht und daher auch keine Treiber dafür existieren.
Für exotische oder mit wenig Speicher bestückte Grafikkarten bleibt schließlich noch der Mono-Server, der auch Hercules-Karten unterstützt.

Busmäuse, der Firmen Microsoft, Logitech oder ATI können ebenso wie die üblichen, seriellen Mäuse der bekannten Hersteller oder ein PS/2-kompatibler Trackball verwendet werden.

Bussystem (ISA / EISA)

Mittlerweile werden sowohl Motherboards mit dem gewöhnlichen AT-Bus (ISA) als auch dem flexibleren und schnellern EISA-Bus unterstützt. Auch die inzwischen sehr verbreiteten Local Bus Erweiterungen lassen sich unter Linux betreiben.
Ein EISA-Rechner mit entsprechender EISA oder Local Bus Grafikkarte und EISA SCSI Controller ist derzeit wohl die schnellste Hardware für Linux.

Rechner mit Micro-Channel (MCA) von IBM werden nur teilweise unterstützt, da die Spezifikationen von IBM wohl relativ restriktiv gehandhabt werden, und daher noch Probleme mit dem Festplatten-Interface bestehen.

Zusatzhardware

Streamer

Zur Datensicherung mit Streamern (Bandlaufwerken) konnte man zunächst nur SCSI-Geräte verwenden. Inzwischen gibt es jedoch auch Treiber für preisgünstigere Floppy Streamer, die an einen Disketten- bzw. Kombicontroller angeschlossen werden können.

Netzwerkkarten

Die Auswahl der Treiber für Netzwerkkarten ist recht groß. Neben häufig verwendeten Karten wie NE2000 oder SMC (früher auch WD), werden auch einige 3Com und DLink Karten unterstützt. Viele Noname Karten sind zu den obigen kompatibel und können ebenfalls benutzt werden.

Multi Serial Adapter

Wie es für ein UNIX-System traditionell üblich ist, kann man auch bei Linux ASCII-Terminals verwenden, die an serielle Schnittstellen bzw. spezielle Multi Serial Adapter angeschlossen werden. Auch für solche Karten existieren Treiber.

ISDN

Ein Feature, das Linux in Zukunft von vielen anderen Systemen abheben könnte, ist die direkte Unterstützung von ISDN-Karten im Kernel. An diesem Feature wird von einigen Entwicklern gearbeitet.

Soundkarten

Auch Multimedia-Anwendungen sind unter Linux möglich. Entsprechende Peripherie wie Soundkarten (Soundblaster, Soundblaster 16, Adlib, Gravis Ultra Sound oder PAS 16) oder CD-ROM Laufwerke können über entsprechende Kernel-Treiber angesprochen werden. Auch hier sollte man jedoch auf der jeweils aktuellen Hardwareliste nachsehen, für welche CD-ROM-Laufwerke inzwischen Treiber-Software existieren. Völlig problemlos sind auch hier SCSI-Laufwerke einzubinden.

Transputer

Sogar für eher exotische Hardware, wie Transputer Boards, existieren Treiber, die es ermöglichen, diese Karten unter Linux anzusprechen.

In Hinblick auf die weitere Entwicklung im Hardware und Softwarebereich ist derzeit ein 486er Rechner ab 33 MHz und mit

16 MB Hauptspeicher eine vernünftige und erschwingliche Ausstattung. Die Festplatte sollte mindestens 200 MB besitzen, da permananter Speicherplatzmangel ein sinnvolles Arbeiten unmöglich macht. Ein 14-Zoll-Monitor ist eigentlich zu klein für eine Multitasking-Umgebung, da meist viele Fenster gleichzeitig geöffnet werden. Der Linux X-Server ist in der Lage, einen größeren, virtuellen Bildschirm zur Verfügung zu stellen und die meisten Geräte können eine Auflösung von 800 mal 600 Punkten vernünftig darstellen; deshalb kann für den Einstieg auf einen 17- Zoll-Monitor verzichtet werden.

5.4 Installation des Systems

Im folgenden wird der Installationsvorgang am Beispiel des SLS-Paketes näher beschrieben. Dieser Vorgang ist jedoch mit geringeren Abweichungen auch auf andere Installationspakete übertragbar.

Die wesentlichen Schritte einer Linux-Installation sind :

- Booten des Systems von Diskette
- Erstellen der Partitionen
- Anlegen der Filesysteme
- Kopieren des Systems auf die Festplatte
- Anpassung der Systemdateien
- Installation des Bootmanagers
- Konfiguration der grafischen Oberfläche
- Einrichten der Benutzer

Bootdiskette

In der Regel beginnt der Installationsvorgang mit dem Booten einer Linux-Installationsdiskette. Auf dieser Diskette ist ein minimales Linux-System mit wenigen Utilities und einem Installationsscript enthalten.
Der Inhalt dieser Installationsdiskette liegt meist als Datei vor. Um aus dieser Datei eine bootfähige Diskette zu erstellen, muß

sie mit einem speziellen Utility auf die Diskette übertragen werden. Unter DOS gibt es dazu ein Programm mit dem Namen `rawrite`, auf einem UNIX-System kann der Befehl `dd` verwendet werden. Auf eine so erstellte Bootdiskette kann nicht mehr mit normalen DOS-Befehlen zugegriffen werden.
Im SLS-Paket gibt es zwei Dateien für Installationsdisketten. Eine für 3,5-Zoll-Disketten mit dem Namen `a1.3` und eine für 5,25- Zoll-Disketten mit dem Namen `a1.5`. Slackware kennt nur eine derartige Diskette namens `bootdisk`.
Nach dem Booten der Installationsdiskette muß man sich mit der Benutzerkennung *root* oder *install* einloggen. Als Benutzer *install* wird sofort das SLS-Installationsscript gestartet, von dem aus alle weiteren Schritte in einem Menü ausgewählt werden können. Loggt man sich als root ein, befindet man sich in einer UNIX-Shell und kann die folgenden Schritte selbst durchführen. Slackware stellt ein Installations-Script namens `setup` zur Verfügung , das die im fogenden beschriebenen Schritte weitgehend automatisch durchführt.
Um bei eventuellen Problemfällen schnell eine Lösung zu finden, ist jedoch ein etwas tieferes Verständnis für die ablaufenden Vorgänge notwendig. Daher wird im folgenden der manuelle Installationsvorgang ausführlich beschrieben.

Partitionieren

Zur Installation von Linux benötigt man mindestens eine freie Partition. Es ist jedoch sinnvoll, gleich drei Partitionen anzulegen. Eine für das Root-Filesystem, eine für das Home-Filesystem und eine als Swap-Partition.
Falls auf der Festplatte noch keine Partitionen angelegt sind, die für Linux verwendet werden können, so muß dies mit dem Programm `fdisk` durchgeführt werden. Das SLS-Installationsscript bietet dazu einen Menüpunkt an.

```
dirk1:/root# fdisk /dev/hdb

Command (m for help): p

Disk /dev/hdb: 15 heads, 17 sectors, 1001 cylinders
Units = cylinders of 255 * 512 bytes

   Device Boot  Begin   Start     End  Blocks Id  System

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-1001): 1
Last cylinder or +size or +sizeM or +sizeK (1-1001): +100M

Command (m for help): p

Disk /dev/hdb: 15 heads, 17 sectors, 1001 cylinders
Units = cylinders of 255 * 512 bytes

   Device Boot  Begin   Start   End  Blocks   Id  System
/dev/hdb1           1       1   804  102509+  83  Linux extfs

Command (m for help): w
The partition table has been altered.
Please reboot before doing anything else.
dirk1:/home/root#
```

Anlegen einer Partition mit fdisk unter Linux

Zu beachten ist dabei, daß eine Swap-Partition nicht größer als 16 MB sein kann, falls der Kernel älter ist als 0.99PL14. Mehr wird vom Linux Kernel nicht verwaltet. Es ist jedoch möglich, bei sehr hohem Speicherbedarf mehrere Swap-Partitionen oder Swap-Files anzulegen.

Man sollte einer Linux-Partition den Typ 82 (Linux Native) und einer Swap-Partition den Typ 83 zuweisen. Dieser hat zwar keinen Einfluß auf die Funktion, schafft jedoch Klarheit bei der Zuordnung einer Partition.

Alle Partitionen, auch die logischen Laufwerke einer extended Partition, werden einfach durchnumeriert und können normal verwendet werden. Sie werden auf Dateien im `/dev`-Verzeichnis, deren Name mit `hd` bzw. `sd` für SCSI-Festplatten beginnt, abgebildet. Genaueres hierzu ist im Kapitel über Grundlagen zu finden.

Anlegen der Filesysteme

Bevor auf einer Partition Dateien gespeichert werden können, muß auf ihr ein Filesystem angelegt werden. Im allgemeinen wird

dazu der Befehl `mkfs` verwendet. Unter Linux sollte ein Extended2 Filesystem verwendet werden, da dieses gegenüber dem normalen MINIX-Filesystem schneller ist. Zur Erzeugung wird der Befehl `mke2fs` benützt. Als Parameter genügt es, den Pfad des Devices, das für die gewünschte Partition steht, zu übergeben.

```
dirk1:/root# mke2fs -c /dev/fd0
mke2fs 0.2c+, 93/03/22 for EXT2 FS alpha 0.2b, 93/03/01
Block Log Size = 0
Fragment Log Size = 0
360 inodes
1440 blocks
72 blocks reserved for root
First data block=1 (1)
Block size=1024
1 blocks group
8192 blocks per group
8192 fragments per group
360 inodes per group
dirk1:/root#
```

Anlegen eines Filesystems am Beispiel einer Diskette

Mit dem SLS-Installationsscript muß dieser Befehl ebenfalls nicht eingegeben werden, sondern kann in einem Menü ausgewählt werden.

Anlegen der Swap Partition

Ähnlich wie ein Filesystem wird eine Swap Partition mit dem Befehl `mkswap` vorbereitet. Dazu muß jedoch neben dem Device die Anzahl der Blöcke der Partition angegeben werden. Diese erhält man am einfachsten durch Aufruf des Programms `fdisk` und Eingabe von **<p>.** Damit läßt sich die aktuelle Partitionstabelle ausgeben. In dieser Tabelle ist unter anderem die Größe jeder Partition in Blöcken angegeben.

```
dirk1:/root# mkswap /dev/hda3 8000
Setting up swapspace, size = 18192000 bytes
dirk1:/home/root#
```

Vorbereiten einer Swap-Partition

Wie bei den Filesystemen muß auch dieser Befehl nicht manuell eingegeben werden, wenn man Linux mit dem SLS-Installations-

script installiert. Alle nötigen Schritte können hier aus einem Menü ausgewählt werden.

Kopieren auf die Festplatte

Sind die Filesysteme und die Swap-Partitionen eingerichtet, so kann man das Linux-System auf die Festplatte kopieren.
Beim SLS-Paket wählt man dazu den entsprechenden Punkt aus dem Installationsmenü aus oder ruft direkt das Installationsscript `doinstall` auf. Slackware erlaubt ebenfalls, eine Auswahl der zu installierenden Komponenten zu treffen.
Danach werden einige Angaben vom Benutzer abgefragt. Dazu gehört das Medium, von dem aus installiert werden soll sowie der Umfang der Installation. Zur Auswahl stehen verschiedene Varianten, wobei das komplette SLS-Paket derzeit ca. 90 MB benötigt.
Anschließend werden die Disketten der Reihe nach auf das neue Filesystem kopiert.

Ist dieser Schritt beendet, wird man aufgefordert, eine leere Diskette einzulegen, auf die das Kernel Image kopiert wird. Von dieser Diskette kann später das neu installierte System gebootet werden. Anschließend werden nur noch einige allgemeine Systemparameter abgefragt, die benötigt werden, um symbolische Links anzulegen und eine Grundeinstellung der Konfigurationsdateien durchzuführen.
Außerdem kann der Linux Loader (LILO) automatisch installiert werden, mit dem Linux von der Festplatte gestartet werden kann. Diese automatische Installation ist jedoch nicht immer problemlos und sollte daher erst zu einem späteren Zeitpunkt durchgeführt werden.
Damit ist die Installation abgeschlossen und der Rechner kann neu gebootet werden.

5.5 Der Bootmanager (LILO)

Der Linux Loader (LILO) ist ein Programm, mit dem Linux sofort beim Booten geladen werden kann. Befinden sich mehrere Betriebssysteme auf der Festplatte, so kann die Auswahl des zu startenden Systems auch über LILO erfolgen. Neben Linux können hiermit auch DOS, OS/2 und andere PC–UNIX Varianten gestartet werden. Es ist sogar möglich, diese von einer zweiten Festplatte zu booten.
Das Funktionsprinzip ist ähnlich dem des OS/2-Bootmanagers. Anstatt sofort ein Betriebssystem zu laden, wird zunächst LILO gestartet, der alle angemeldeten Systeme und Konfigurationen zur Auswahl anbietet.

Bedienung

Der Loader meldet sich zunächst mit einem "LILO " auf dem Bildschirm. Dann hat der Benutzer eine einstellbare Anzahl von Sekunden Zeit, eine der Tasten **<Alt>**, **<Strg>** oder **<AltGr>** zu drücken um damit dem Loader mitzuteilen, daß man nicht die Standardkonfiguration booten, sondern eine andere Partition oder Konfiguration auswählen möchte. Daraufhin wird der Benutzer durch die Ausgabe der konfigurierbaren Meldung `boot:` aufgefordert, eine Boot-Variante einzugeben.
Sobald der Boot-Prompt erscheint, kann durch Drücken der Tabulatortaste eine Liste der verfügbaren Alternativen ausgegeben werden.

```
LILO <Alt>
LILO boot:<Tab>
linux   linux-old   dos
boot:linux

Loading Linux
```

Der Linux Loader beim Booten

Wird beim Booten innerhalb einer einstellbaren Zeit keine Taste gedrückt, so wird automatisch die erste Auswahl getroffen und damit das erste definierte Betriebssystem geladen.

Installation

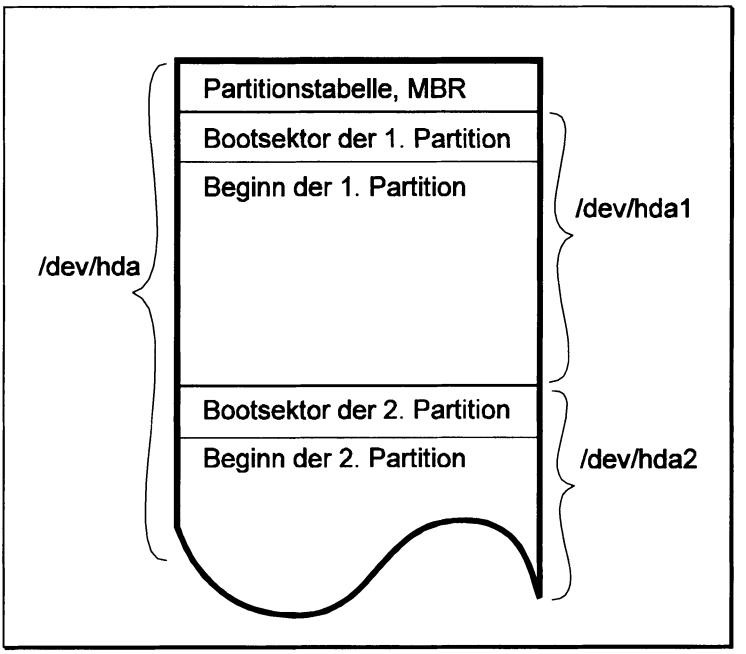
LILO wird normalerweise im Verzeichnis `/etc/lilo` installiert. Dort stehen neben dem eigentlichen LILO-Programm auch die Konfigurationsdateien und ein README File, in dem die Installation und Konfiguration ausführlich beschrieben wird.

Im folgenden wird eine typische Installation für ein System mit 2 AT-Bus-Festplatten beschrieben, wobei Linux auf der zweiten Festplatte installiert wurde.
Zunächst wird die Datei `/etc/lilo/config` angepaßt.

```
boot = /dev/hda
install = /etc/lilo/boot.b
delay = 100
compact

image = /vmlinuz
        label = linux
        root = /dev/hdb1
        vga=1

image = /vmlinuz.old
        label = linux-old
        root = /dev/hdb1
        vga=1

other = /dev/hda1
        label = DOS
```

Beispiel einer Konfigurationsdatei für den Linux Loader

Die erste Zeile gibt an, wo LILO installiert werden soll. Zur Auswahl steht der Anfang einer Festplatte und damit der *Master Boot Record* (MBR) oder der Anfang einer Partition. Am unkompliziertesten ist es, LILO direkt in den MBR der ersten Festplatte zu installieren. Damit wird LILO auch automatisch als erstes gestartet, unabhängig davon, welche Partition in der Partitionstabelle als aktiv gekennzeichnet ist.
Vorsicht! Die Installation von LILO im Master Boot Record führt dazu, daß der ursprüngliche DOS-MBR überschrieben wird. Dieser kann jedoch mittels `fdisk` auch unter DOS wiederhergestellt werden.
Falls LILO zusammen mit einem anderen Bootmanager, z.B. dem von OS/2 verwendet werden soll, ist dies jedoch nicht sinnvoll (siehe unten). In diesem Fall wird LILO am Anfang einer Partition installiert.



Beispiel für den Aufbau der ersten Festplatte

`install = /etc/lilo/boot.b` gibt die Bootdatei an und wird für die Standard Installation benutzt.
`delay` gibt die Zeit in zehntel Sekunden an, die LILO warten soll, ob eine der oben genannten Tasten gedrückt wird, ehe automatisch die erste Boot-Alternative ausgewählt wird.
`compact` ist eine Optimierung von LILO, die verhindert, daß jeder Sektor einzeln gelesen wird.
Der Eintrag `image = /vmlinuz` und die folgenden eingerückten Zeilen beschreiben die erste Auswahlmöglichkeit. Das Kernel Imagefile, das gebootet werden soll, heißt `vmlinuz` und steht im Root-Verzeichnis. Die Kennung, die beim Booten eingegeben werden muß ist `linux`.
Die zweite Boot-Möglichkeit ist das File `/vmlinuz.old`. Es ist ein älteres Kernel Imagefile das für Notfälle vorgesehen ist, in denen der aktuelle Kernel nicht mehr funktioniert.
Die dritte Alternative ist das Booten von der DOS-Partition auf der ersten AT-Bus-Festplatte. Die Marke `other` führt dazu, daß nicht versucht wird, von dieser Partition aus Linux sondern den Loader eines anderen Betriessystem zu starten.

Kernel Imagefiles

Ein Kernel Image ist eine Datei, die den Kernel, also den eigentlichen Betriebssystemkern zusammen mit einem Initialisierungs- und Ladeprogramm enthält. Diese Datei wird erstellt, wenn der Kernel, dessen Quellcodes in `/usr/src/linux` stehen, kompiliert wird (siehe Konfiguration und Compilation des Kernels).
Es bietet sich an, wenn man einen neuen Kernel übersetzt hat, die alte Version ebenfalls in der Konfigurationsdatei einzutragen. Falls man einen Fehler bei der Compilation oder Konfiguration des Kernels gemacht hat, und das System mit dem neuen Kernel nicht mehr bootet, kann man jederzeit mit dem alten Kernel booten und den Fehler korrigieren.
Dies wird auch schon von dem Makefile des Kernels unterstützt. Beim Aufruf von `make zlilo` im Verzeichnis des Kernel-Quellcodes `/usr/src/linux` wird nach Abschluß der Compilation das neue Kernel Imagefile nach `/vmlinuz` kopiert, und das alte `/vmlinuz` in `/vmlinuz.old` umbenannt.

Aktivieren des Loaders

Damit die Einträge in dieser Konfigurationsdatei auch aktiv werden, muß LILO neu installiert werden. Dazu wird das Script `install` im Verzeichnis `/etc/lilo` aufgerufen, das den eigentlichen Loader mit der aktuellen Konfiguration in einen Bootsektor oder das MBR schreibt.
Vorsicht! Da es auch ein normales Kommando `install` gibt, muß das Script `install` des LILO explizit mit seinem Pfad aufgerufen werden, also `/etc/lilo/install` beziehungsweise `./install`, falls man sich schon im Verzeichnis `/etc/lilo` befindet.

```
dirk1:/etc/lilo# ./install
Added linux
Added linux-old
Added DOS
dirk1:/etc/lilo#
```

Aktivieren der Einstellungen des Linux Loaders

Entfernen des LILO

Hat man LILO in den MBR installiert und möchte ihn aus irgendwelchen Gründen wieder entfernen, so muß der alte Inhalt des MBR wieder hergestellt werden.
Falls man DOS auf einer anderen Festplatte oder Partition installiert hat, ist die einfachste Möglichkeit dazu, unter DOS das Programm `fdisk` mit der Option `/mbr` aufzurufen. Dadurch wird ein neuer Master Boot Record installiert und damit LILO überschrieben.

Alternativer Bootmanager

Möchte man anstelle des LILO einen anderen Bootmanager verwenden, wie zum Beispiel den OS/2-Bootmanager, so darf man LILO nicht im MBR installieren, da dieser sonst in jedem Fall als erstes gestartet wird. Statt dessen bietet es sich an, LILO am Anfang der Linux Root Partition zu installieren.
Zur Partitionierung sollte in diesem Fall die OS/2-Version von `fdisk` benutzt werden. Anschließend kann der OS/2-Bootmanager initialisiert und die Bootpartition aktiviert werden. Nun wird auch die Linux Partition angemeldet, auf der zuvor der Linux Loader installiert wurde.
Beim Booten wird nun zunächst der OS/2-Bootmanager aktiviert, aus dem heraus der Benutzer dann entweder OS/2 oder über LILO das Linux-System starten kann.
Weitere Informationen zu Bootmanagern können dem *LILO User's Guide* entnommen werden.

Konfiguration

Sind die Systemdateien auf die Festplatte kopiert und der Linux Loader installiert, so sollten noch einige Konfigurationsdateien angepaßt werden. Dadurch wird das System auf die vorhandene Hardware angepaßt.

6.1 Allgemeine Konfiguration

Die meisten Konfigurationsdateien sind im Verzeichnis `/etc` abgelegt. Viele dieser Dateien müssen normalerweise nicht geändert werden. Ihre kurze Beschreibung befindet sich im Anhang. Im folgenden werden nur die Einstellungen behandelt, die nach der Installation auf einer normalen PC-Hardware verändert werden müssen.

Filesysteme

Beim Starten des Systems wird aus einem der `rc`-Scripts (siehe Kapitel *Administration*) der Befehl `mount -a` aufgerufen, der alle Filesysteme aus der Datei `/etc/fstab` in den Verzeichnisbaum einbindet.
In dieser Datei stehen alle verfügbaren Dateisysteme zusammen mit ihren Optionen, die beim Mounten als Parameter an den `mount`-Befehl übergeben werden.
Hat man bei der Installation, wie im vorausgehenden Kapitel beschrieben, mehrere Filesysteme angelegt, so müssen diese in der Datei `/etc/fstab` eingetragen werden. Das folgende

Beispiel zeigt die Einträge für einen Rechner, auf dem zwei Dateisysteme eingerichtet wurden.

```
/dev/sda2       /               ext2    defaults
/dev/sda5       /home           ext2    defaults
none            /proc           proc    defaults
```

Ausschnitt einer Datei /etc/fstab

Der Eintrag, der das `/proc`-Filesystem mountet, ist wichtig, da einige Kommandos, wie zum Beispiel `ps`, von diesem abhängig sind. Wie schon im Kapitel *Grundlagen* erläutert wird hier Systeminformation in Form von Dateien und Unterverzeichnissen dargestellt.

Das `/home`-Verzeichnis sollte sich auf einem eigenen Dateisystem befinden. So kann im Bedarfsfall das System neu installiert werden, ohne daß wichtige Benutzerdaten verloren gehen. Falls vor dem Mounten eines Filesystem auf dieses Verzeichnis bereits Dateien oder Unterverzeichnisse vorhanden sind, so müssen sie vorher entfernt werden. Es bietet sich an, diese temporär auszulagern und nach dem Mounten wieder zu restaurieren.

```
mount /dev/sda3 /mnt
cp -r /home/* /mnt
rm -rf /home/*
umount /mnt
mount /dev/sda3 /home
```

Beispielhafte Befehlsfolge zum Mounten des Home-Filesystems

Natürlich sollte die `/home`-Partition auch in die Datei `/etc/fstab` eingetragen werden, so daß sie beim nächsten Systemstart automatisch eingebunden wird.

Swapspace

Falls man acht MB oder weniger Hauptspeicher besitzt, ist es nötig eine Swap-Partition oder eine Swap-Datei anzulegen, falls dies bei der Installation noch nicht erfolgt ist. Der Hauptspeicher reicht sonst nicht aus, um mehrere Programme gleichzeitig unter der grafischen Oberfläche ablaufen zu lassen.

Vier MB reichen nicht einmal um den Kernel neu zu kompilieren und daneben einen Editor zu benutzen. Aber auch bei 16 MB ist es sinnvoll, den virtuellen Speicher durch eine Swap-Partition zu vergrößern.
Der Swapbereich wird mit dem Befehl `swapon` aktiviert. Damit dies automatisch beim Systemstart geschieht, wird er wie ein Filesystem in der Datei `/etc/fstab` eingetragen.
Eine Datei `/etc/fstab`, in der sowohl normale Filesysteme als auch eine Swap-Partition und NFS Mounts eingetragen sind, könnte zum Beispiel folgendermaßen aussehen :

```
/dev/sda2        /                ext2    defaults
/dev/sda5        /home            ext2    defaults
none                  /proc       proc    defaults

/dev/sda3        /                swap    defaults

sun1:/home     /sun               nfs     defaults
linux1:/               /linux1    nfs     defaults
```

Beispiel einer Datei /etc/fstab

Die Einträge entsprechen weitgehend den Parametern des `mount`-Kommandos. In der ersten Spalte wird das einzubindende Device angegeben. Im Falle eines NFS-Mounts steht hier der Host und, durch einen Doppelpunkt getrennt, das entsprechende Verzeichnis. In der zweiten Spalte findet sich eine Pfadangabe, die die Stelle im Dateisystem angibt, an der die Dateien erscheinen sollen. Die dritte Spalte enthält den Typ des Dateisystems. Die meisten Dateisysteme erlauben die Angabe zusätzlicher Optionen, die in der letzten Spalte übergeben werden. Sind keine weiteren Parameter erwünscht, so sollte hier in jedem Fall `defaults` stehen.

Login

Viele Optionen, die mit dem Einloggen von Benutzern zu tun haben, werden in der Datei `login.defs` im `/etc`-Verzeichnis eingestellt. Hier können beispielsweise die Wartezeit nach Eingabe eines falschen Passwortes und die Devices, von denen man sich als *root* einloggen kann, angegeben werden.

Entsprechende Hinweise stehen in der Datei `/etc/login.defs` selbst. Die folgende Abbildung zeigt einen kleinen Ausschnitt aus dieser Datei.

```
# Pause in Sekunden, die der Login Prompt nach einem
# falschen Passwort gesperrt ist.
FAIL_DELAY      2

# Devices von denen ein Login als root zulässig ist
#CONSOLE        /etc/consoles
#CONSOLE        console:tty01:tty02:tty03:tty04
CONSOLE tty1:tty2:tty3:tty4:tty5:tty6:tty8

#  Dateien, die nach dem Einloggen ausgegeben werden
MOTD_FILE       /etc/motd
```

Ausschnitt aus einer Datei /etc/login.defs

Tastaturanpassung

Seit der Version 0.99.10 des Linux Kernels muß die Anpassung der Tastatur nicht mehr bei der Compilation des Kernels erfolgen, sondern kann zur Laufzeit geändert werden. Dazu gibt es den Befehl `loadkeys`, der eine Tabelle mit einem Tastaturlayout (Map) läd.

Dieser Befehl sollte beim Starten des Systems aus einer der `rc`-Dateien aufgerufen werden. Als Tastaturtabelle sollte man die Datei `gr-latin1.map` verwenden, die eine normale deutsche Tastatur mit Umlauten nach ISO Latin-1 definiert.

Der Befehl `loadkeys` und die Map-Dateien sind beim SLS-Paket als Quellcode im Verzeichnis `/usr/src` enthalten. Das Kommando `loadkeys` sollte nach der Compilation in das `/etc`- und die verschiedenen Tabellen in das `/usr/lib/keymaps`-Verzeichnis kopiert werden.

Um nicht bei jedem Neustart die Tastaturtabelle neu laden zu müssen, sollte ein entsprechendes Kommando in der `/etc/rc`-Datei ergänzt werden.

```
# Länderspezifische Tastaturtabelle laden
/etc/loadkeys /usr/lib/keymaps/gr-latin1.map
```

Laden der deutschen Tastaturanpassung

6.2 Der Kernel

Der Kernel, also der eigentliche Kern von Linux, ist wie auch die Kommandos, Utilities und alle anderen Linux-Programme im Quellcode frei verfügbar. Er ist hauptsächlich in C geschrieben, kleine Teile auch in Assembler.
Im Kernel sind neben dem Scheduler, der für das Umschalten der verschiedenen Prozesse zuständig ist, vor allem die Treiber für die Peripheriegeräte und die Routinen zur Verwaltung der Dateisysteme enthalten.

Konfiguration des Kernels

Ein weiterer Schritt nach der Basisinstallation des Betriebssystems ist die Konfiguration und Compilation des Kernels. Dies kann jedoch in vielen Fällen entfallen. Um eine optimale Anpassung an die verfügbare Hardware zu erreichen, ist eine Übersetzung jedoch unbedingt zu empfehlen. Dabei können Treiber, die nicht benötigt werden, weggelassen, neue Treiber dazugenommen oder Einstellungen von Treibern geändert werden. Dadurch wird der Speicherbedarf des Kernels geringer, und der Bootvorgang läuft schneller ab.

Treiber

Normalerweise befinden sich die Quelltexte des Kernels im Verzeichnis `/usr/src/linux`.
Dort findet man auch ein Konfigurationsscript, das vom Makefile aufgerufen wird und die individuelle Anpassung erheblich erleichtert. Diese wird durch Eingabe von `make config` aktiviert und erlaubt unter anderem die Einstellung folgender Optionen:

- Unterstützte Filesysteme
- TCP/IP Unterstützung
- Verwenden des Koprozessor Emulators
- Optimierung für 486er Prozessoren
- SCSI Unterstützung
- Treiber für SCSI und Netzwerkkarten
- Parameter für spezielle Interface-Karten
- Einstellungen für Soundkarten

Compilation

Hat man alle nötigen Angaben gemacht, so müssen die Abhängigkeiten der einzelnen Quelltexte untereinander neu bestimmt werden. Dieser Vorgang wird durch den Aufruf von `make dep` gestartet. Auch dazu muß man sich im Hauptverzeichnis des Kernel-Quellcodes (`/usr/src/linux`) befinden. Da bei einer Änderung der Konfiguration eventuell noch alte Objekt-Dateien vorhanden sein könnten, sollten diese mittels `make clean` gelöscht werden.
Dann kann der eigentliche Compilationsvorgang gestartet werden. Das Makefile spielt dabei, wie bei der Installation und Compilation der meisten anderen Programme, eine zentrale Rolle. Es steuert, welche Quelldateien übersetzt werden müssen, und erkennt, welche Module schon übersetzt wurden. Außerdem dient es zur Koordination verschiedener Scripts zur Konfiguration und Installation des Kernels.
Mittlerweile wird standardmäßig ein komprimierter Kernel erzeugt. Dieser wird beim Booten dann durch eine integrierte Dekompressions-Routine entpackt. Diese Eigenschaft des Linux-Kernels ist auch der Grund dafür, daß ein komplettes Linux-Minimalsystem mit Unterstützung sämtlicher Hardware (Netzwerk, Streamer, SCSI-Geräte, CD-ROM) noch auf eine einzige Bootdiskette paßt. Der zeitliche Verlust ist dabei nicht weiter von Bedeutung.
Im folgenden werden die wichtigsten Varianten beim Aufruf des `make`-Kommandos aufgelistet.

- **`make dep`** - Die Abhängigkeiten der Quelldateien werden neu bestimmt. Dies sollte nach jeder Änderung der Konfiguration des Kernels gemacht werden.
- **`make clean`** - Löscht alle Objektdateien. Beim nächsten Compilieren wird alles neu übersetzt.
- **`make`** - Der Kernel wird compiliert und ein neues Kernel Image wird im Verzeichnis `/usr/src/linux` mit dem Namen `zImage` erstellt. Dieser Vorgang läuft auch bei vielen anderen Aufrufen ab, die dann dieses Imagefile weiterverwenden.

- **make zlilo** - Erstellt wie oben ein neues Kernel Iimage und kopiert es nach `/vmlinuz`. Eine eventuelle vorhandene alte, gleichnamige Datei wird vorher nach `/vmlinuz.old` gesichert. Danach wird das Installationsscript des Linux Loaders `/etc/lilo/install` aufgerufen, so daß beim nächsten Systemstart der neue Kernel gebootet werden kann.
- **make disk** - Damit wird am Ende der Compilation das Imagefile direkt auf die Diskette, die gerade im Laufwerk A liegt, geschrieben. Man sollte also schon vor dem Aufruf dieses Kommandos eine leere Diskette einlegen.

6.3 Daemons

Einige der Daemons, die beim Booten des Systems gestartet werden, haben Einstellungen, die in Konfigurationsdateien abgelegt sind.
Da den Daemons jedoch kein Terminal zugeordnet ist, geben sie in der Regel keine Fehlermeldungen aus, sondern schicken alle Fehlermeldungen und anderen Hinweise als Nachricht an den Syslog Daemon.
Um eventuelle Fehler bei der Konfiguration der Daemons und anderer Programme einfacher erkennen zu können, ist es daher sinnvoll, zunächst den Syslog Daemon zu konfigurieren.

Syslog Daemon

Der Syslog Daemon `syslogd` kann die Meldungen, die er von anderen Daemons bekommt, in eine Datei schreiben, per E-Mail an bestimmte Benutzer senden oder direkt auf den Bildschirm oder die Konsole ausgeben.
Diese Einstellung kann individuell für die Einheit, von der die Meldung kommt und für die Priorität der Meldung vorgenommen werden. Sie wird in der Datei `/etc/syslog.conf` festgelegt.
Eine meist ausreichende Lösung ist die, alle Meldungen einer bestimmten Dringlichkeit in einer Datei zusammenzufassen. Die Einträge, die dazu in `/etc/syslog.conf` notwendig sind, sehen dann wie folgt aus:

```
*.alert          /var/log/alert
*.emerg          /var/log/emerg
*.crit           /var/log/crit
*.err            /var/log/err
*.warning        /var/log/warning
*.notice         /var/log/notice
*.info           /var/log/info
*.debug          /var/log/debug
```

Basiskonfiguration des Syslog Daemons

Alle Einträge bestehen aus einer Einheiten- und Prioritätsangabe und dem Ziel der Meldungen. Eine Übersicht über alle definierten Einheiten und Prioritäten findet man in der Manual page zu `syslogd` oder zur Datei `syslog.conf`. Diese hat unter Linux allerdings den Namen `syslconf`.
Die Dateien im Verzeichnis `/var/log` müssen beim Start des Syslog Daemons bereits existieren. Zur Erzeugung einer leeren Datei benutzt man am besten das Kommando `touch` mit der entsprechenden Datei als Parameter. Um zu überprüfen, ob die Einstellungen in der Konfigurationsdatei korrekt sind, kann man den Syslog Daemon mit dem `kill`-Befehl beenden und ihn danach mit der Option `-d` neu starten. Diese Option startet den Daemon im Debug Modus, wodurch dieser in einer Matrix auf dem Bildschirm ausgibt, welche Meldungen in welche Datei geschrieben oder an welchen Benutzer geschickt werden.
Da in der obigen Konfiguration alle Meldungen immer an Dateien angefügt werden, sollte man darauf achten, daß diese Dateien nicht zu groß werden. Dazu bietet es sich an, in einem Script, das von `crond` regelmäßig aufgerufen wird, die protokollierten Dateien in ein anderes Verzeichnis zu sichern und dann neu anzulegen. Anschließend muß dem Syslog Daemon noch mittels `/etc/syslogd.reload` mitgeteilt werden, seine Mitteilungsdateien neu zu öffnen. Auf diese Weise kann man bei Problemen immer noch in den alten Protokolldateien nachsehen.

Drucker-Daemon

Die allgemeine Definition und Konfiguration der Drucker erfolgt in der Datei `/etc/printcap`. Darin wird beispielsweise festgelegt, ob für jeden neuen Druckerjob ein eigenes Deckblatt erzeugt wird, oder ob nach der Ausgabe eine Leerseite nachgeschoben werden soll. Außerdem ist es möglich, mehrere Druckerwarteschlangen zu erzeugen, für die wiederum verschiedene Filterprogramme angemeldet werden können. Um den Zugriff auf Drucker anderer Rechner (Printserver) über das Netz zu realisieren, muß der eigene Rechner zusätzlich in der Datei `/etc/hosts.lpd` des Printservers eingetragen werden. Diese Datei enhält die Namen aller Hosts, die auf den Drucker zugreifen können.
Das folgende Beispiel zeigt die Datei `/etc/printcap` eines Linux-Rechners, der keinen eigenen Drucker, dafür jedoch Zugriff auf den Drucker einer Workstation hat.

```
ibmdr|risc1lpr|ibm 4019::lp=:rm=risc1:\
        sd=/usr/spool/ibmdr:lf=/usr/spool/ibmdr/ibmdr-log:
ibmps|risc1ps|ibm 4019 Postscript::lp=:rm=risc1:\
       rp=ps:sd=/usr/spool/ibmps:lf=/usr/spool/ibmps/ibmps-log:
```

Beispiel einer Datei /etc/printcap

Eine Liste der Optionen, die in dieser Datei angegeben werden können, findet man in der Manual page zu `printcap`.

Umlaute

Ein Problem bei der Ausgabe von Textdateien auf einen Drucker stellen meistens die deutschen Umlaute dar. Eine befriedigende Lösung kann hier wohl nur durch die Installation eines entsprechenden Filters erzielt werden. Ein derartiger Filter kann auf viele verschiedene Arten realisiert werden. Im folgenden wird ein einfaches C-Programm aufgelistet, das neben der Konvertierung der Umlaute auch noch nach jeder Zeile ein Steuerzeichen für den Wagenrücklauf (CR) an den Drucker ausgibt. Alternativ könnte auch das UNIX-Kommando `tr` zur Zeichenkonversion herangezogen werden.

```
/*****************************************************************
 *
 * Umlaut-Konvertierung für EPSON Drucker
 *
 *****************************************************************/

#include <stdio.h>

main (int argc, char *argv[])
{
  int ch;

  while ((ch = getchar ()) != EOF)
  {
    /* printer needs CR+LF */
    if (ch == '\n')
      putchar ('\r');

    /* convert ISO to PC */
    switch (ch)
    {
      case 228: /* "a */
        ch = 132;
        break;
      case 246: /* "o */
        ch = 148;
        break;
      case 252: /* "u */
        ch = 129;
        break;
      case 196: /* "A */
        ch = 142;
        break;
      case 214: /* "O */
        ch = 153;
        break;
      case 220: /* "U */
        ch = 154;
        break;
      case 223: /* sz */
        ch = 225;
        break;
      case 167: /* paragraph */
        ch = 21;
        break;
      default:
        break;
    }
    putchar (ch);
  }
}
```

Filter zur Umlaut-Konvertierung

Die Einbindung eines solchen Filters erfolgt über die Option `if` oder `of` in der `/etc/printcap`-Datei. Ein `of`-Filter wird im Gegensatz zu einem `if`-Filter nur einmal initialisiert. Letzterer wird für jeden Druckerjob erneut gestartet.

```
#
#       /etc/printcap
#
lp:lp=/dev/lp1:sf:sd=/usr/spool/lp:mx=0:sh

# Text Warteschlange
txt:lp=/dev/lp1:sf:sd=/usr/spool/txt:\
      if=/usr/spool/lp/epson:mx=0:sh

# Postscript Warteschlange
ps:lp=/dev/lp1:sf:sd=/usr/spool/ps:\
     if=/usr/spool/lp/Postscript:mx=0:sh
```

Einbindung eines Filters in `/etc/printcap`

Durch die Anmeldung mehrerer Druckerwarteschlangen (printer queue) kann bei Bedarf zwischen den einzelnen Filtern umgeschaltet werden. Die Auswahl der richtigen Queue erfolgt über einen Parameter des `lpr`-Kommandos.

```
linux1:/home/tul> lpr -Ptxt Umlaut.txt
```

Ausdruck eines Textes mit Umlauten

Postscript

Über einen passenden Filter läßt sich ein gewöhnlicher Nadel-, Tintenstrahl- oder Laserdrucker problemlos in einen vollwertigen Postscript-Drucker verwandeln. Dazu wird einfach der Postscript-Interpreter *Ghostscript* als Filter angemeldet, was jedoch nicht direkt, wohl aber über ein Shell Script erfolgen kann.

```
#!/bin/bash
exec /usr/bin/gs -q -sPAPERSIZE=a4 -dSAFER\
     -sDEVICE=epson -sOutputFile=- -
```

Postscript-Filter

Im obigen Beispiel wird angenommen, daß es sich bei dem anzusteuernden Drucker um ein EPSON-kompatibles Gerät handelt. Durch die Anpassung des `sDevice` Parameters können aber auch andere Druckertypen benutzt werden.
Findige Linux-Anwender haben sich mittlerweile intelligente Shell Scripts ausgedacht, die automatisch erkennen sollen, um welche Art von Dateiformat es sich handelt und dann den richtigen Filter aufrufen. Da eine derartige Erkennung aber nicht immer eindeutig funktioniert, soll hier nicht näher darauf eingegangen werden.

6.4 Streamer und CD-ROM

SCSI

Viele PCs verfügen heute über ein Streamer- oder CD-ROM-Laufwerk. Auch Linux untertstützt derartige Massenspeicher. Im Falle einer Neuanschaffung sollte man sich in jedem Fall für ein SCSI-Gerät entscheiden. Die Anpassung von SCSI-Geräten gestaltet sich aufgrund der Standardisierung des Befehlssatzes extrem einfach. Es genügt, die entsprechenden Einträge im `/dev`-Verzeichnis über das Kommando `mknod` zu erzeugen, falls der zugehörige SCSI-Treiber bereits in den Kernel compiliert wurde. Die meisten Distributionen legen diese Device-Dateien schon bei der Installation automatisch an.
Soll beispielsweise ein CD-ROM-Laufwerk (scd?) und ein SCSI-Streamer (rmt?) angesprochen werden, so müssen die beiden folgenden Einträge existieren:

```
ls scd* rmt*
crw-rw-rw-   1 root     root       9,   0 Jan 23  1993 rmt0
crw-rw-rw-   1 root     root       9,   1 Jan 23  1993 rmt1
brw-rw-rw-   1 root     root      11,   0 Jan 23  1993 scd0
brw-rw-rw-   1 root     root      11,   1 Jan 23  1993 scd1
linux1:/dev>
```

Device-Dateien für SCSI-Streamer und CD-ROM-Laufwerke

Ist dies nicht der Fall, so können diese mit den folgenden Befehlen vom Systemadministrator generiert werden:

```
linux1:/dev>mknod /dev/rmt0 c 9 0
linux1:/dev>mknod /dev/scd0 b 11 0
```

Erzeugung von Device-Dateien über das Kommando mknod

Erheblich problematischer kann die Konfiguration eines Floppy-Streamers oder eines CD-ROM-Laufwerkes mit eigenem AT-Bus-Controller sein, da für diese Geräte kein Standard für entsprechende Treiber existiert. Mittlerweile sind jedoch für die gängigsten Modelle auch unter Linux passende Treiber verfügbar, die ebenfalls in den Kernel compiliert werden müssen.

6.5 Netzkonfiguration

Wer Zugriff auf ein TCP/IP-Netzwerk hat, sei es an einer Hochschule mit Anschluß an das Internet, einer Firma oder zu Hause, der kann seinen Linux-PC an dieses anbinden. Die dazu notwendigen Schritte werden in den folgenden Abschnitten näher beschrieben.

Adressen

Bei einer Netzwerkverbindung mit TCP/IP bekommt jede Netzwerkschnittstelle eines Rechners eine weltweit eindeutige IP- Adresse zugewiesen. Sie besteht aus vier Zahlen zwischen 0 und 255, die durch Punkte getrennt sind, also zum Beispiel 141.7.1.40. Diese Adresse muß beim Starten des Rechners in einem der `rc`-Scripte im `/etc`-Verzeichnis eingestellt werden.

IP-Adresse

Außerdem ist es wichtig zu wissen, wie diese IP-Adresse sich in die Netzwerkadresse und die Hostadresse aufteilt.
Man spricht hier von der Netzwerkmaske. Andere wichtige Daten sind die Adresse des Routers und eventuell die Adresse des Nameservers. Die Details dieser Verwaltung zu diskutieren würde den Rahmen dieses Buches sprengen. Deshalb werden im folgenden nur auf die praktischen Auswirkungen dieser Parameter eingegangen und auf die vielen Bücher über TCP/IP und Netzwerkadministration verwiesen, die sich ausschließlich diesem Thema widmen.

Netzwerkmaske

Bei größeren bestehenden Netzen gibt es meist einen Netzwerkadministrator, der die Adressen verwaltet und auch alle weiteren Parameter kennt.
Für kleine private Netzwerke, die keine Verbindung zu anderen Netzen haben, ist die IP-Adresse relativ unerheblich. Man kann sie frei wählen oder die vom Installationsprogramm voreingestellte IP-Adresse beibehalten. Wichtig ist nur, daß sie unter den angeschlossenen Rechnern eindeutig ist.
Als Netzwerkmaske verwendet man am einfachsten 255.255.255.0, also ein Netz der Klasse C, ohne Subnetze.

Netzwerkklasse

Man unterscheidet im allgemeinen drei Klassen von Netzen (A-C), die jeweils eine unterschiedliche Anzahl von Bytes zur Addressierung benötigen (1-3). Die Netzwerkadresse erhält man durch logische Und-Verknüpfung der IP-Adresse und der Netzwerkmaske, bei der IP-Adresse 141.7.1.20 also 141.7.1.0. Die Broadcast-Adresse ist im einfachsten Fall nur die Netzwerkadresse mit einer abschließenden 255 statt der 0.

Broadcast-Adresse

Für andere Rechner in diesem Netz sollte man nur die letzte Zahl der IP-Adresse verändern und nur Zahlen zwischen 2 und 253 verwenden, da die 1 oder die 254 meist für Router reserviert sind und 0 oder 255 der Netzwerk- bzw. Broadcast-Adresse zugeordnet werden.

Loopback Device

Auch bei Rechnern, die eigentlich keine Netzwerkschnittstelle besitzen, existiert eine virtuelle Schnittstelle mit dem Namen *Loopback*. Wie der Name schon andeutet, wird alles, was über diese Schnittstelle ausgegeben wird, direkt wieder eingelesen. Damit sind TCP/IP-Verbindungen des Rechners mit sich selbst möglich. Man kann also auch ohne Netzwerkschnittstelle die TCP/IP-Programme benutzen. Die IP-Adresse der Loopbackschnittstelle ist in der Regel 127.0.0.1.

Konfiguration der Schnittstelle

ifconfig

Die Konfiguration der Netzwerkschnittstellen eines Rechners und damit die Festlegung der IP-Adresse und der anderen oben genannten Parameter erfolgt über den `ifconfig`-Befehl. Als Parameter werden die zu konfigurierende Schnittstelle, die IP-Adresse sowie weitere optionale Parameter angegeben. Details zu diesen Parametern findet man in der Manual page zu `ifconfig`. Die Namen der Schnittstellen werden beim Bootvorgang von den Treibern ausgegeben. Üblich sind zum Beispiel `eth0` für Ethernet, `lo0` für Loopback oder `sl0` für die erste serielle Schnittstelle bei SLIP.

Routing

Sind die Schnittstellen konfiguriert, müssen noch Einträge in die interne Routing-Tabelle des Kernels gemacht werden, in der gespeichert wird, welche Adressen über welche Schnittstellen erreichbar sind. Dies wird mit dem Befehl `route` durchgeführt. Das folgende Beispiel zeigt das Script `/etc/rc.inet1`, das die nötigen Einstellungen für einen Rechner mit der IP-Adresse 141.7.1.40 im Netz 141.7.1.0 durchführt.

route

```
# Loopback konfigurieren
/etc/ifconfig lo 127.0.0.1

# Ethernet konfigurieren
/etc/ifconfig eth0 141.7.1.40 broadcast 141.7.1.255 netmask
255.255.255.0

# routes eintragen
/etc/route add 127.0.0.1
/etc/route add 141.7.1.40
/etc/route add default gw 141.7.1.254
```

Konfiguration der Netzwerkschnittstellen beim Booten

SLIP

SLIP-Verbindungen werden normalerweise nicht beim Systemstart konfiguriert, sondern erst dann, wenn sie benötigt werden. Meist verwendet man das SLIP-Protokoll zum Aufbau einer TCP/IP-Verbindung über eine Modem-Strecke. Die Konfiguration erfolgt also erst mit der erfolgreichen Anwahl des SLIP-Servers.

Modem

Zum Wählen der Telefonnummer und Aufbauen der Verbindung zu einem SLIP-Router oder Server gibt es das Programm `dip`. Es kann interaktiv bedient oder von einem Script gesteuert werden. Nachdem die Verbindung mit `dip` hergestellt ist, steht eine konfigurierte SLIP-Schnittstelle zur Verfügung, über die sofort Daten übertragen werden können.

Das folgende Beispiel ist ein einfaches Script zum Anwählen eines SLIP-Routers. Es wird mit `dip <scriptname>` gestartet.

```
port cua1
speed 2400
reset
wait OK 4
dial 123456
wait CONNECT 60
set $remote 141.7.19.1
set $local  141.7.19.10
mode SLIP
```

Beispiel eines `dip` Scripts

Nachdem die Verbindung aufgebaut ist, kann mit `route add default gw 141.7.19.1` der Standard-Routing-Eintrag für diese SLIP-Verbindung gesetzt werden. Alle Rechner, die über den SLIP-Router erreicht werden können, sind dann direkt ansprechbar.

Ping

Das wohl wichtigste Hilfsprogramm zur Konfiguration eines UNIX-Netzwerkes ist das Kommando `ping`. Nachdem die Netzwerkschnittstelle mittels `ifconfig` aktiviert wurde und die notwendigen Routing-Informationen eingetragen wurden, kann die Verbindung geprüft werden. `Ping` sendet in regelmäßigen Abständen kleine Datenpakete an den Zielrechner, der diese dann beantwortet.

```
linux0:/home/tul> ping linux1
PING linux1.openconcepts.com (192.0.2.130): 56 data bytes
64 bytes from 192.0.2.130: icmp_seq=0 ttl=119 time=2 ms
64 bytes from 192.0.2.130: icmp_seq=1 ttl=120 time=1 ms
64 bytes from 192.0.2.130: icmp_seq=2 ttl=121 time=1 ms
64 bytes from 192.0.2.130: icmp_seq=3 ttl=122 time=1 ms
64 bytes from 192.0.2.130: icmp_seq=4 ttl=123 time=1 ms

--- linux1.openconcepts.com ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 1/1/2 ms
linux0:/home/tul>
```

Das `ping` kommando

Auf diese Weise ist auch eine Beurteilung der Übertragungsgeschwindigkeit möglich. Erst wenn die Verbindung mittels `ping` erfolgreich getestet wurde, kann mit der weiteren Netzwerkkonfiguration fortgefahren werden.

Hostname und Nameserver

Die direkte Angabe der IP-Adresse ist für den Anwender meist nicht sehr komfortabel. Daher wurde eine Möglichkeit geschaffen, alternativ symbolische Bezeichner zu benutzen. Eine solche symbolische Adresse besteht aus dem Host- und dem Domainnamen.

Der Hostname ist eine lokal eindeutige Bezeichnung für eine Maschine. Der Domainname dagegen ist international eindeutig. Statt der IP-Adresse `141.7.1.40` kann beispielsweise auch `linux1.rz.fh-heilbronn.de` angegeben werden. In diesem Fall wäre `linux1` der Hostname und `rz.fh-heilbronn.de` der Name der Domain. Die Zuordnung zwischen IP- und symbolischer Adresse erfolgt in der Datei `/etc/hosts`.

Hostname

Hier kann optional auch eine Abkürzung festgelegt werden, die im lokalen Netz die Eingabe langer symbolischer Adressen erspart.

```
#IP Adresse    symbolische Adresse           Abkürzung

141.7.1.40     linux1.rz.fh-heilbronn.de     linux1

127.0.0.1      localhost
127.0.0.1      loopback

141.7.1.20     sun1.rz.fh-heilbronn.de       sun1
141.7.1.25     risc11.rz.fh-heilbronn.de     risc1
```

Ausschnitt aus der Datei /etc/hosts

Da es nicht möglich ist, hunderte oder tausende von Einträgen in dieser Datei zu warten, wurde ein System von hierarchisch aufgebauten Nameservern eingerichtet, die diese Namen verwalten und automatisch austauschen. Für jede Domain, also alle Rechner mit einem bestimmten Domainnamen, gibt es einen zuständigen Nameserver.

Nameserver

Die UNIX TCP/IP-Programme wandeln den Hostnamen durch Aufruf einer C-Library-Routine in die IP-Adresse um. Diese Routine, die auch *Resolver* genannt wird, greift auf die Datei `/etc/hosts` zu und stellt eventuell eine Verbindung zum nächsten Nameserver her. In welcher Reihenfolge dies geschieht kann in der Datei `/etc/host.conf` eingestellt werden.

Resolver

```
order hosts bind
multi on
```

Die Datei /etc/host.conf

Der Eintrag `order hosts bind` gibt an, daß zunächst in der Datei `/etc/hosts` nachgesehen werden soll. Falls dort kein Eintrag zum gesuchten Hostnamen vorhanden ist, wird der Nameserver kontaktiert. Die Adresse des Nameservers wird in der Datei `/etc/resolv.conf` abgelegt.
In vielen Linux-Installationspaketen, unter anderem im SLS-Paket ist ein Nameserver enthalten. Seine Konfiguration ist mit dem SLS-Installationsscript recht einfach. Für kleinere Netze ist ein Nameserver jedoch meist überflüssig.
Eine genauere Beschreibung der Funktionsweise des Nameservers und der TCP/IP-Konfigurationsdateien findet man zum Beispiel im *Linux Network Administration Guide* (NAG) oder in anderen Büchern über TCP/IP.

E-Mail Konfiguration

Die Konfiguration des `smail` Daemons, der für das Senden und Empfangen von Nachrichten zuständig ist, erfolgt über die Dateien im Verzeichnis `/usr/lib/smail`. Im einfachsten Fall genügen folgende Einträge in der Datei `config`.

```
#
#  Visible domain name
#
visible_domain=rz.fh-heilbronn.de
#
# Visible host name
#
visible_name=linux1.rz.fh-heilbronn.de
```

Einfache `smail` Konfiguration

Genauere Infomationen zur Konfiguration findet man vor allem in der zu `smail` gehörigen Manual page oder im *Linux Network Administration Guide* (NAG).

Inetd

`Inetd` ist der Daemon, der die Internet Services verwaltet. Er wartet auf Verbindungsanforderungen mit den Portnummern, die für die einzelnen Services festgelegt sind und startet den zuständigen Daemon erst dann, wenn tatsächlich ein Verbindungswunsch vorliegt.
Die Tabelle, in der angegeben ist, welcher Daemon für einen bestimmten Service zuständig ist, wird mit dem Namen `/etc/inetd.conf` bezeichnet.

```
telnet  stream  tcp  nowait  root  /etc/telnetd    telnetd
ntalk   dgram   udp  wait    root  /etc/ntalkd     ntalkd
ftp     stream  tcp  nowait  root  /etc/ftpd       ftpd -l
finger  stream  tcp  nowait  root  /etc/fingerd    finger
shell   stream  tcp  nowait  root  /etc/rshd       rshd
login   stream  tcp  nowait  root  /etc/rlogind    rlogind
tftp    dgram   udp  wait    root  /etc/tftpd     tftpd /home/ftp
# Internal to inetd
echo    stream  tcp  nowait  root  internal
echo    dgram   udp  wait    root  internal
discard stream  tcp  nowait  root  internal
discard dgram   udp  wait    root  internal
daytime stream  tcp  nowait  root  internal
daytime dgram   udp  wait    root  internal
chargen stream  tcp  nowait  root  internal
chargen dgram   udp  wait    root  internal
```

Die Datei /etc/inetd.conf

Die Zuordnung der Services zu Portnummern steht in der Datei `/etc/services`.

```
tcpmux          1/tcp       # TCP Port Service Multiplexer
rje             5/tcp       # remote job entry
echo            7/tcp
echo            7/udp
discard         9/tcp       sink null
discard         9/udp       sink null
systat          11/udp      users
systat          11/tcp      users
daytime         13/udp
daytime         13/tcp
daytime         13/udp
netstat         15/udp
netstat         15/tcp
qotd            17/udp      quote
quote           17/tcp      # quote of the day
chargen         19/tcp      ttytst source
chargen         19/udp      ttytst source
ftp-data        20/tcp
ftp             21/tcp
telnet          23/tcp
smtp            25/tcp      mail #Simple Mail Transfer
nsw-fe          27/tcp      # NSW User System FE [24, RHT]
```

Ausschnitt aus einer Datei /etc/services

Diese Dateien werden normalerweise nicht geändert. Sie müssen nur dann modifiziert werden, wenn neue Services hinzugefügt werden oder bei einem Update eines Daemon neue Optionen nötig sind.

Berkeley r-Utilities

hosts.equiv, hosts.lpd

Auch die Berkeley r-Utilities `rlogin`, `rsh` und `rcp` besitzen eigene Konfigurationsdateien. Um von einem Rechner über diese Programme Zugriff auf einen anderen erhalten zu können, muß der Client-Rechner in der Datei `/etc/hosts.equiv` des Servers eingetragen sein. Reicht der Zugriff auf eine Druckerwarteschlange, so genügt es, den Namen des Clients in `/etc/hosts.lpd` einzutragen. Es ist aber auch möglich, einzelnen Benutzern anderer Maschinen Zugriffrechte einzuräumen. In diesem Fall wird zusätzlich die Benutzerkennung angegeben.

```
#
# Zulässige Hosts und Benutzer
#

linux1.rz.fh-heilbronn.de
linux2.rz.fh-heilbronn.de strobel
sun1.rz.fh-heilbronn.de          arnold
```

Die Dateien `hosts.equiv`, `hosts.lpd` und `.rhosts`

.rhosts

Die Dateien `/etc/hosts.equiv` und `/etc/hosts.lpd` können nur vom Systemadministrator modifiziert werden. Möchte ein einzelner Benutzer einem anderen Zugriffrechte einräumen, so kann er dies über einen Eintrag in die Datei `.rhosts` in seinem Home-Verzeichnis erreichen. Eine solche Datei ist vor allem dann sinnvoll, wenn ein Anwender auf mehreren Maschinen im Netz unterschiedliche Benutzerkennungen besitzt. Durch eine entsprechend angepaßte `.rhosts`-Datei wird der netzweite Zugriff auf die eigenen Verzeichniss erheblich erleichtert.

NFS und Mount Daemons

Das Network File System (NFS) benötigt mehrere Daemons. Da NFS auf Remote Procedure Calls (RPC) basiert, wird zunächst der Portmap Daemon benötigt. Er registriert die RPC-Dienste eines Servers und liefert die TCP/IP-Portnummer an anfragende Clients.
Die Anfragen bezüglich NFS von einem Client-Rechner werden vom NFS Daemon beantwortet. Seine wichtigste Konfigurationsdatei ist `/etc/exports`, in der alle Verzeichnisse zusammen mit den Zugriffsrechten aufgelistet sind, die von anderen Rechnern via NFS gemountet werden dürfen.
Für die Mount-Information selbst ist der Daemon `mountd` zuständig. Er verwaltet Verzeichnisse und beantwortet Mount-Anfragen.

```
#
# Exportierte Verzeichnisse
#
/               linux1(rw)
/home           141.7.1.49(rw)
/home/prog      risc1(rw)
```

Beispiel einer Datei /etc/exports

6.6 X11-Konfiguration

Die Installation des X-Windows Systems unter Linux besteht normalerweise nur aus dem Entpacken der Programme und Dateien aus den verschiedenen tar-Archiven. Bei den meisten Paketen geschieht dies schon während der Installation des Betriebssystems und stellt so keinerlei Problem dar.
Anders verhält es sich bei der Konfiguration, bei der der X-Server auf die verwendete Videokarte und den Monitor abgestimmt werden muß. Dies geschieht durch Modifikation der zentralen Konfigurationsdatei `/usr/lib/X11/Xconfig`.

Die Xconfig-Datei

In dieser Datei werden zunächst die vom X-Server benötigten Pfade definiert. Dabei handelt es sich um den Zugriffspfad auf die RGB-Farbtabellen oder auf die Font-Verzeichnisse. Diese Einstellungen sind bei den Installationspaketen meist schon korrekt vorgenommen.

```
# Datei mit Farbdefinitionen
RGBPath         "/usr/X386/lib/X11/rgb"

# ZUgriffspfade für Zeichensätze
FontPath    "/usr/lib/X11/fonts/Type1/"
FontPath    "/usr/lib/X11/fonts/75dpi/"
FontPath    "/usr/lib/X11/fonts/Speedo/"
FontPath    "/usr/lib/X11/fonts/BitstreamType1/"
FontPath    "/usr/lib/X11/fonts/misc/"

# Tastatur
Keyboard
  AutoRepeat 500 5
#  Xleds      1 2 3
  ServerNumLock
#  DontZap

LeftAlt         Meta
RightAlt        ModeShift
ScrollLock      ModeLock

# Maus
Microsoft       "/dev/mouse"
#MouseSystems   "/dev/mouse"
#MMSeries       "/dev/mouse"
#Logitech       "/dev/mouse"
#MouseMan       "/dev/mouse"
#Busmouse       "/dev/mouse"
```

Ausschnitt aus einer Xconfig-Datei

In Anschluß daran werden die Funktionen spezieller Modifier-Tasten definiert. Zu beachten ist hier, daß die rechte **<Alt>** Taste zu `ModeShift` definiert werden muß, damit die Sonderzeichen der deutschen Tastatur wie `"@"` mit dieser Taste angesprochen werden können.
In vielen Beispieldateien, die im gleichen Verzeichnis vorliegen, wird hier von einer amerikanischen Tastaturbelegung ausgegangen und die **<AltGr>** Taste als `Compose`-Taste verwendet.
Danach erfolgt die Anpassung der Maus, was durch Angabe des Typs und der verwendeten Schnittstelle erfolgt.
Anschließend kommen die Einstellungen für die VGA-Karte, wie Chipsatz und verfügbare Pixeltaktfrequenzen bzw. Quarze (Clocks). Bei den einfachen Karten kann diese Angabe entfallen,

da der Chipsatz beim Starten des X-Servers erkannt wird, und die Quarze ebenfalls ermittelt werden.

```
vga256
  Virtual       1024 768
  ViewPort      0 0
  Modes         "640x480"  "800x600" "1024x768i"
  Chipset       "et4000"
  Videoram      1024
  Clocks        25 28 32 36 40 48 50 65  0 57  0 72 80
```

Definition der VGA-Karte in der XConfig-Datei

Es ist jedoch sinnvoll, diese dennoch anzugeben, da der ermittelte Zahlenwert als Bezeichner verwendet wird, auf den sich später die Definitionen der Videomodi beziehen. Falls nun bei der Ermittlung der verfügbaren Pixeltaktfrequenzen Schwankungen auftreten und ein Quarz zum Beispiel anstelle von 49.5 als 50 erkannt wird, so kann es vorkommen, daß der X-Server beim Starten den verwendeten Pixeltakt eines Videomodus nicht identifiziert, und mit einer Fehlermeldung abbricht.

Ermitteln der Pixeltaktfrequenzen (Clocks)

Um die vorhandenen Frequenzen zu ermitteln, kann man die Clocks-Zeile aus der `Xconfig`-Datei entfernen und den X-Server starten. Dieser gibt dann die gefundenen Clock-Werte im Textmodus aus, wo man sie durch Zurückschalten in den Textmodus mit **<Strg-Alt>** und der Nummer der virtuellen Konsole sehen kann.

In der Zeile, die mit `Modes` beginnt, werden die Namen der Videomodi aufgelistet, die zur Laufzeit verfügbar sein sollen. Während der X-Server schon läuft, kann dann mit den Tasten **<Strg-Alt +>** und **<Strg-Alt ->** des Nummernblocks die Auflösung zwischen diesen Modi umgeschaltet werden, die jedoch noch genauer definiert werden müssen.

Einstellen der Videomodi

Der schwierigste und gefährlichste Teil der Konfiguration ist das Einstellen der Videomodi, da hier direkt die Ablenkfrequenzen des Monitors definiert werden, und ein billiger Monitor, der keine Schutzschaltung gegen zu hohe Ablenkfrequenzen besitzt, durch falsche Werte zerstört werden kann. Die der X386-Distribution

beiliegende `Xconfig`-Datei enthält aber schon die wichtigsten Standardmodi, die auf allen heutigen Monitoren funktionieren sollten, ohne Schaden anzurichten.
Der Vorteil dieser Art der Konfiguration des Videomodus ist, daß ein vorhandener Monitor optimal ausgenutzt werden kann. So ist es zum Beispiel möglich, einen 14" Monitor, dessen maximale horizontale Ablenkfrequenz zu gering ist, um 800 x 600 Bildpunkte flimmerfrei darzustellen, in einer Auflösung von 800 x 550 mit 72 Hz Bildwiederholfrequenz zu betreiben.
In der Datei `/usr/lib/X11/Xconfig` werden für jeden Modus der zu verwendende Clock, sowie vier Werte für das horizontale Timing und vier Werte für das vertikale Timing angegeben.

```
# Mode      Clock      horizontal                vertical
 "800x600"   45        800  840 1030 1184       600  600  606 624
```

Definition eines Videomodes

8 Werte beschreiben den Videomodus

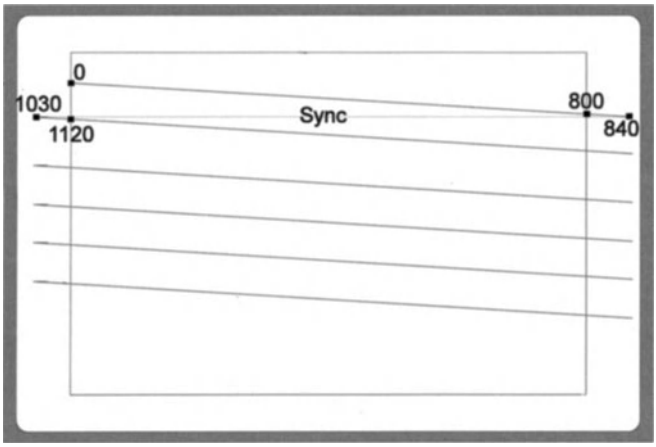
Die Werte bedeuten der Reihe nach

- die maximale Anzahl der Pixel, nach denen kein Bild mehr angezeigt wird,
- die Anzahl der Pixelschritte bis zum Anfang des horizontalen Synchronisationsimpulses (Sync), wobei die Werte immer weitergezählt werden,
- die Anzahl der Pixelschritte bis der Sync beendet ist, und die zweite Beruhigungsphase des Strahls beginnt,
- die gesamte Anzahl der Pixelschritte bis ein Zyklus beendet ist.

```
Modes        "800x5" "800x600"
Clocks      25 28 45 40 33 29
Chipset      "et3000"
Videoram    512

ModeDB

#              clock  horizontal timing      vertical timing
 "800x600"     40     800  800  860 1030     600  604  610  624
 "800x5"       45     800  840 1030 1120     540  540  546  558
```

In diesem Beispiel werden drei Modi definiert. Die Karte besitzt einen ET3000 Chip und 512 KB RAM. Die Clocks werden fest

definiert und bekommen Namen, so daß der Quarz mit dem Namen 45 die dritte verfügbare Pixeltaktfrequenz aktiviert.
Die Zeile mit "800x5" bedeutet, daß ein Modus mit dem Namen "800x5" definiert wird, für den der Quarz mit dem Namen 45 verwendet werden soll.



schematische Darstellung des Bildaufbaus

Die horizontale Auflösung beträgt 800 Pixel, und nach dem Ende der sichtbaren Zeile beginnt eine Pause, in der sich der Strahl beruhigen kann. Die Pause geht bis 840, danach beginnt der Synchronisationsimpuls. Dieser dauert 190 Pixelschritte, bis 1030 und danach ist eine Pause bis 1120, nach der der nächste horizontale Zyklus beginnt.
Nach 540 horizontalen Zyklen, also Zeilen, kommt ohne Pause die vertikale Synchronisierung, die 6 horizontale Zyklen dauert, und nach der wieder eine Pause bis zum 558. Zyklus ist. Danach kommt das nächste Bild.

Ermittlung der Zahlenwerte

Die genauen Regeln zur Ermittlung der einzelnen Werte für einen solchen Videomodus werden ausführlich in der Datei `/usr/lib/X11/doc/video.tutorial` beschrieben. Oft ist es jedoch einfacher, in den vielen Beispieldateien einen passenden Eintrag zu suchen und diesen abzuändern.
Außerdem gibt es ein Rechenblatt für die im SLS-Paket enthaltene einfache Tabellenkalkulation `xspread`, das die Berechnung der Werte vereinfacht.

Der begrenzende Faktor bei einfacheren Monitoren ist in der Regel die maximale horizontale Ablenkfrequenz. Darunter versteht man die Frequenz, mit der der Elektronenstrahl von links nach rechts und von Zeile zu Zeile bewegt wird. Man berechnet diese Frequenz, durch Teilen des Pixeltakts, der in Mhz angegeben ist, durch die größte (rechte) Zahl des Blocks für das horizontale Timing.

$$f_{horizontal} = \frac{f_{pixel}}{N_{pixel}}$$

Berechnung der horizontalen Ablenkfrequenz

Am obigen Beispiel wäre die erforderliche horizontale Ablenkfrequenz 45 Mhz / 1120, also ca. 40 kHz. Das ist die obere Grenze für den im Beispiel verwendeten Monitor.

Bildwiederholfrequenz

Um daraus die Bildwiederholfrequenz zu ermitteln, teilt man die soeben ermittelte horizontale Ablenkfrequenz durch die Anzahl der Zeilen (also horizontalen Zyklen), die ein kompletter Durchlauf benötigt. Dies ist die rechte Zahl des Blocks für das vertikale Timing.

$$f_{vertikal} = \frac{f_{horizontal}}{N_{Zeilen}}$$

Berechnung der Bildwiederholfrequenz

Am obigen Beispiel 40 kHz / 558 also 72 Hz.

Flimmern des Bildschirms

Werden anstelle der 540 Zeilen 600 Zeilen dargestellt, so sinkt die Bildwiederholfrequenz deutlich unter 72 Hz, was man als leichtes Flimmern des Bildschirms bemerkt.

Bei einem besseren Monitor, dessen maximale horizontale Ablenkfrequenz zum Beispiel bei 60 kHz liegt, und einer neueren Grafikkarte, die einen schnelleren Clock anbietet, könnte man

den Pixeltakt erhöhen, und so eine höhere Bildwiederholfrequenz erreichen.

Um einen vorhandenen Videomodus abzuändern, empfiehlt es sich, den Modus mehrfach zu kopieren und abzuändern und die geänderten Modi jeweils mit einem anderen Namen in die Modes Zeile einzutragen. Dann kann man den X-Server starten und durch Umschalten mit **<Strg-Alt>** und der + oder - Taste des Nummernblocks die Auswirkungen der Änderungen vergleichen. Falls der Monitor bei einem neuen Modus nicht mehr synchronisiert, also kein stabiles Bild mehr anzeigt, sollte man den Modus schnell wechseln oder den X-Server mit **<Strg-Alt-Backspace>** beenden, um eine Beschädigung des Monitors zu vermeiden.

Tastaturanpassung

Die Verwaltung der Tastatur erfolgt unter X-Windows unabhängig vom Kernel. Standardmäßig wird eine amerikanische Tastatur initialisiert. Länderspezifische Anpassungen können aber in einer Datei namens `.Xmodmap` durchgeführt werden. Soll eine systemweite Definition des Tastaturlayouts erfolgen, so muß die Datei im Verzeichnis `/usr/lib/X11/xinit` stehen. Diese kann aber auch von jedem Benutzer selbst modifiziert werden, wenn sie sich in dessen Home-Verzeichnis befindet. Die Aktivierung eines neuen Tastaturlayouts geschieht über das Kommando `xmodmap`, das der Benutzer auch direkt aufrufen kann.

Komfortabler ist jedoch mit Sicherheit die Verwendung des `xkeykaps`-Utilities, das einigen Linux-Distributionen beiliegt, und auf den üblichen FTP-Servern zu finden ist. Dieses Programm verfügt über eine X11-Oberfläche. Der Benutzer kann sich auf einfache Weise die aktuelle Tastaturbelegung anzeigen lassen und interaktiv mit der Maus verändern.

xkeykaps

Interaktive Umbelegung der Tastatur mit `xkeykaps`

In den amerikanischen Linux-Distributionen fehlen oftmals deutsche Tastaturanpassungen. Daher soll hier ein Beispiel für eine deutsche `.Xmodmap` gegeben werden. Dabei ist auf das Tastatursymbol Nummer 12 zu achten, das meist fälschlicherweise mit dem Symbol `paragraph` an Stelle von `section` belegt wird.

```
keycode   8 =
keycode   9 = Escape
keycode  10 = 1 exclam
keycode  11 = 2 quotedbl twosuperior
keycode  12 = 3 section threesuperior
keycode  13 = 4 dollar
keycode  14 = 5 percent
keycode  15 = 6 ampersand
keycode  16 = 7 slash braceleft
keycode  17 = 8 parenleft bracketleft
keycode  18 = 9 parenright bracketright
keycode  19 = 0 equal braceright
keycode  20 = ssharp question backslash
keycode  21 = apostrophe grave
keycode  22 = BackSpace
keycode  23 = Tab
keycode  24 = q Q at
keycode  25 = W
keycode  26 = E
keycode  27 = R
keycode  28 = T
keycode  29 = Z
keycode  30 = U
keycode  32 = O
keycode  33 = P
keycode  34 = Udiaeresis
keycode  35 = plus asterisk asciitilde
keycode  36 = Return
keycode  37 = Control_L
keycode  38 = A
keycode  39 = S
keycode  40 = D
keycode  41 = F
keycode  42 = G
keycode  43 = H
keycode  44 = J
keycode  45 = k K Arabic_kaf
keycode  46 = l L Arabic_lam Greek_lambda
keycode  47 = Odiaeresis
keycode  48 = Adiaeresis
keycode  49 = asciicircum degree
```

```
keycode  50 = Shift_L
keycode  51 = numbersign apostrophe
keycode  52 = Y
keycode  53 = X
keycode  54 = C
keycode  55 = V
keycode  56 = B
keycode  57 = N
keycode  58 = m M mu
keycode  59 = comma semicolon
keycode  60 = period colon
keycode  61 = minus underscore
keycode  62 = Shift_R
keycode  63 = KP_Multiply
keycode  64 = Alt_L
keycode  65 = space
keycode  66 = Caps_Lock
keycode  67 = F1
keycode  68 = F2
keycode  69 = F3
keycode  70 = F4
keycode  71 = F5
keycode  72 = F6
keycode  73 = F7
keycode  74 = F8
keycode  75 = F9
keycode  76 = F10
keycode  77 = Num_Lock
keycode  78 = Scroll_Lock
keycode  79 = Home KP_7 KP_7 Home
keycode  80 = Up KP_8 KP_8 Up
keycode  81 = Prior KP_9 KP_9 Prior
keycode  82 = KP_Subtract
keycode  83 = Left KP_4 KP_4 Left
keycode  84 = Begin KP_5 KP_5 Begin
keycode  85 = Right KP_6 KP_6 Right
keycode  86 = KP_Add
keycode  87 = End KP_1 KP_1 End
keycode  88 = Down KP_2 KP_2 Down
keycode  89 = Next KP_3 KP_3 Next
keycode  90 = Insert KP_0 KP_0 Insert
keycode  91 = Delete KP_Decimal KP_Decimal Delete
keycode  92 = 0x1007ff00
keycode  93 =
keycode  94 = less greater bar
keycode  95 = F11
keycode  96 = F12
keycode  97 = Home
keycode  98 = Up
keycode  99 = Prior
keycode 100 = Left
keycode 101 = Begin
keycode 102 = Right
keycode 103 = End
keycode 104 = Down
keycode 105 = Next
keycode 106 = Insert
keycode 107 = Delete
keycode 108 = KP_Enter
keycode 109 = Control_R
keycode 110 = Pause
keycode 111 = Print
keycode 112 = KP_Divide
keycode 113 = Mode_switch
keycode 114 = Break
```

Deutsche X-Windows Tastaturanpassung

Administration

Nachdem die Installation des Linux-Systems abgeschlossen ist und die wichtigsten Konfigurationen durchgeführt wurden, kann auch anderen Benutzern der Zugang zum System ermöglicht werden. Außerdem wird bald der Wunsch nach zusätzlichen Applikationen auftauchen, die nicht in den Installationspaketen enthalten sind. Nach einiger Zeit werden von einzelnen Systemkomponenten neuere Versionen erscheinen, die installiert werden sollten, um mit der weiteren Entwicklung Schritt halten zu können. Die damit verbundenen Aufgaben werden unter dem Begriff *Systemadministration* zusammengefaßt.
Da diese auf allen UNIX-Systemen sehr ähnlich sind, soll hier auf die Standardliteratur verwiesen werden und nur einige Tips und Hinweise zu Linux-spezifischen Details gegeben werden.

7.1 Der Administrator

Die Konfigurationsdateien des Systems können nur von dem Benutzer *root*, also dem Systemadministrator geändert werden. Dies wird durch entsprechende Zugriffsrechte der Dateien sichergestellt. Als Systemadministrator hat man generell Zugriff auf alle Dateien und kann diese beliebig modifizieren. Dies bedeutet natürlich auch, daß man als Administrator durch eine falsche Eingabe das gesamte System zerstören oder löschen kann. Wird zum Beispiel der Befehl

```
linux1:/> rm -rf *
```

zum Löschen aller Dateien eines Verzeichnisses mit allen Unterverzeichnissen ohne Sicherheitsabfrage versehentlich im Root-Verzeichnis (/) ausgeführt, so wird unweigerlich das gesamte System gelöscht.
Geschieht dies ohne Root-Privilegien, so sorgen die Zugriffsberechtigungen der Systemdateien und Verzeichnisse dafür, daß schlimmstenfalls alle Dateien des Benutzers gelöscht werden. Dies hat jedoch keinen Einfluß auf andere Benutzer oder das gesamte System.
Daher sollte man als Systemadministrator sehr vorsichtig vorgehen und sich nur dann als *root* einloggen, wenn tatsächlich eine Systemdatei verändert werden muß oder ein neues Programm installiert werden soll.

7.2 Der Bootvorgang

Zum besseren Verständnis des Systems soll zunächst beschrieben werden, wie der Bootvorgang eines Linux-Systems abläuft und welche Programme und Scripts zu diesem Zeitpunkt abgearbeitet werden.
Unabhängig vom Betriebssystem wird beim Booten zunächst der sogenannte Master Boot Record (MBR) geladen. Dort befindet sich die Partitionstabelle und ein Ladeprogramm, welches den Bootsektor der aktiven Partition lädt.

LILO

Ist LILO, wie im Installationskapitel beschrieben, im MBR installiert, so wird er als erstes gestartet und bietet danach zwei verschiedene Linux Kernels und DOS zur Auswahl an. Wählt der Benutzer einen Linux Kernel aus, so wird das entsprechende Kernel-Image geladen und gestartet. Damit beginnt der eigentliche Start des Linux-Systems.

Kernel

Als erstes wird die Grafikkarte vom Kernel initialisiert und eventuell die zu verwendende Bildschirmauflösung abgefragt. Dann werden die verschiedenen Devicetreiber initialisiert, die meist eine entsprechende Meldung auf dem Bildschirm ausgeben. Ist dies geschehen, wird das Root-Filesystem gemountet und der Kernel startet den Prozeß `init`.
Linux kennt, wie UNIX System V, verschiedene Runlevels, die in der Datei `/etc/inittab` festgelegt werden. Dabei handelt es

sich um verschiedene Konstellationen, in denen nur bestimmte Systemkomponenten aktiviert werden. Normalerweise wird das System im Multiuser-Betrieb hochgefahren. Das bedeutet, daß auf der Konsole und optional den seriellen Schnittstellen mehrere `getty`-Prozesse gestartet werden. Außerdem werden in diesem Modus sämtliche Netzwerk Daemons aktiviert. Alternativ steht ein Single-User-Modus zur Verfügung, der vor allem zur Systemadministration vorgesehen ist. Ein weiterer Runlevel kann beispielsweise statt dem üblichen Terminal-Login einen grafischen Login-Prompt (`xdm`) starten.

Da Init der erste Prozeß ist, den der Kernel startet, hat er immer die Prozeßnummer 1 und ist der Vater aller weiteren Prozesse.
Vom Init-Prozeß wird auch das Script `/etc/rc` ausgeführt. Darin werden verschieden Systemdateien neu initialisiert sowie die lokalen Dateisysteme gemountet. NFS-Dateisysteme werden zunächst noch nicht eingebunden, da das Netzwerk und die entsprechenden Daemons noch nicht gestartet sind.

/etc/rc

Der `mount`-Befehl, der die zusätzlichen Dateisysteme in den Linux-Verzeichnisbaum einhängt, liest die Datei `/etc/fstab`, in der alle Filesysteme mit ihrem Device, Typ und Mountpoint (Verzeichnis, in dem das Filesystem eingehängt wird) aufgelistet sind. Falls irgendwann eine neue Festplatte und neue Filesysteme hinzukommen, sollten diese an dieser Stelle eingetragen werden.
Am Ende der `rc`-Datei wird ein weiteres Script mit dem Namen `rc.local` gestartet.

/etc/fstab

```
#!/bin/sh

# Entfernen der Datein /etc/mtab*
# Sie werden von mount neu angelegt.
/bin/rm -f /etc/mtab*
# Entfernen der Dateien /etc/nologin und /etc/utmp
/bin/rm -f /etc/nologin /etc/utmp
# Anlegen einer neuen Datei /etc/utmp
echo -n "" >> /etc/utmp

# Mounten aller lokalen Filesysteme
/bin/mount -av -t nonfs
# Swap Partitionen aktivieren
/bin/swapon -a

# alte Lockfiles löschen
/bin/rm -f /usr/spool/uucp/LCK*

# Starte rc.local
/bin/sh /etc/rc.local
```

Das Script /etc/rc

Darin werden der Maschinenname festgelegt und verschiedene Hintergrundprogramme (Daemons) gestartet. Falls ein Netzwerkanschluß konfiguriert ist, wird außerdem ein Shell-Script (`rc.net`) zur Initialisierung der Netzwerkumgebung gestartet. Hier werden auch eventuell definierte NFS-Mounts durchgeführt.

7.3 Shutdown

Wie alle UNIX-Systeme darf man einen Linux Rechner nicht einfach ausschalten. Statt dessen muß das System mit dem Befehl `shutdown` definiert heruntergefahren werden.
Dies liegt daran, daß durch den internen Cache des Kernels meist nicht alle Daten, die von Programmen auf die Festplattenschnittstelle geschrieben wurden, tatsächlich schon auf die Festplatte gesichert sind. Hinzu kommt, daß besonders häufig benötigte Information, wie die i-Node-Tabelle oder der Superblock der Dateisysteme ebenfalls im Speicher gehalten werden. Falls der Rechner ohne den Befehl `shutdown` ausgeschaltet wird, so kann es zu Inkonsistenzen auf der Festplatte und Datenverlusten kommen. Der Befehl `shutdown` sorgt dafür, daß alle Puffer vom Speicher auf die Datenträger übertragen werden und alle Prozesse ordnungsgemäß beendet werden. Mit dem Befehl `sync` können die Puffer auf die Festplatte geschrieben werden, ohne daß das System beendet wird. Dieser Befehl wird nur jedoch selten verwendet.

7.4 Der Linux-Verzeichnisbaum

Um neuen Linux-Anwendern und dem noch unerfahrenen Systemadministrator die Orientierung im System zu erleichtern, sollen nun die wichtigsten Verzeichnisse eines typischen Linux-Systems erläutert werden.
Das Verzeichnis `"/"` ist die Wurzel des Linux-Verzeichnisbaumes. Es wird daher auch Wurzel- oder Root-Verzeichnis genannt. In ihm sollten außer den Linux Kernel-Imagefiles, die

zum Booten benötigt werden und den wichtigsten Unterverzeichnissen keine weiteren Dateien vorhanden sein.
Dieses Rootverzeichnis wird häufig als Home-Verzeichnis des Systemadministrators verwendet. Es ist jedoch sinnvoller, dafür ein eigenes Unterverzeichnis, zum Beispiel `/root` zu verwenden. Damit wird es einfacher, zwischen Konfigurationsdateien des Administrators und Systemdateien zu unterscheiden.
Um ein neues Home-Verzeichnis festzulegen reicht es, den entsprechenden Eintrag in der Datei `/etc/passwd` mit einem Editor zu ändern.

Verzeichnisse im Root-Verzeichnis

- **`/etc`** - Das `/etc`-Verzeichnis enthält hauptsächlich System- und Konfigrationsdateien. Dies sind zum Beispiel die Scripte `rc` und `rc.local`, die beim Booten aufgerufen werden oder die Dateien `passwd` und `group` mit der Benutzer- bzw. Gruppeninformation. Daneben werden hier auch häufig Daemons mit ihren jeweiligen Konfigurationsdateien abgelegt.
- **`/etc/lilo`** - In diesem Verzeichnis ist normalerweise der Linux Loader (LILO) installiert.
- **`/etc/skel`** - Die Dateien in diesem Verzeichnis werden beim Anlegen eines neuen Benutzers mit `useradd -m` automatisch in das Home-Verzeichnis des Benutzers kopiert. Hier werden normalerweise Beispiele für benutzerspezifische Konfigurationsdateien abgelegt.
- **`/conf`** - Falls dieses Verzeichnis existiert, sind hier ausschließlich Konfigurationsdateien abgelegt, die sonst in `/etc` oder anderen Verzeichnissen zu finden sind. In diesem Fall existieren statt der eigentlichen Dateien in `/etc` nur Verweise (symbolische Links) auf die Dateien in `/conf` bzw. einem Unterverzeichnis, wie beispielsweise `/conf/net`. In einfachen Installationen wird jedoch meist auf dieses Verzeichnis verzichtet.
- **`/dev`** - Wie der Name `/dev` schon andeutet, liegen in diesem Verzeichnis die Gerätedateien. Das sind spezielle Dateien, die

einem I/O-Treiber zugeordnet sind. (siehe auch Geräte im Kapitel über Grundlagen).

- **/tmp** - Dieses Verzeichnis wird von vielen Programmen für Temporärdateien verwendet. In /tmp kann von allen Benutzern gelesen und geschrieben werden. Dateien, die in diesem Verzeichnis liegen, dürfen in der Regel gelöscht werden, wenn keine Applikations-Prozesse mehr laufen. Außer dem Administrator sollten bei diesem Vorgang aber keine weiteren Benutzer eingeloggt sein. Oftmals wird das Löschen des /tmp-Verzeichnises beim Systemstart durch einen Eintrag in der /etc/rc-Datei durchgeführt.
- **/mnt** - Dieses Verzeichnis sollte leer sein und wird häufig dazu verwendet, Disketten oder Filesysteme anderer Rechner per NFS temporär zu mounten.
- **/user** - Auch dieses Verzeichnis ist in der Regel leer und wird nur zum Mounten verwendet.
- **/proc** - Hier wird normalerweise das Proc-Filesystem gemountet. Das Proc-Filesystem ist ein spezielles Dateisystem, in dem Informationen des Kernels und laufender Prozesse als Unterverzeichnisse und Dateien dargestellt werden. Diese Dateien können meist als Text ausgelesen werden und ermöglichen so einen einfachen Zugriff auf diese Information.
- **/bin** - Die wichtigsten Systemprogramme, die zum Starten des Systems nötig sind oder zur rudimentären Administration der Filesysteme benötigt werden, sind in diesem Verzeichnis abgelegt. Dazu gehören zum Beispiel die Befehle mv, cat, rm oder Programme wie hostname, install, fdisk oder fsck.
- **/lib** - Die Images der Shared Libraries des Systems liegen im Verzeichnis /lib. Das ist der Teil einer Shared Library, der die eigentlichen Routinen enthält, und beim Starten der Programme, die die Library verwenden, geladen wird. Der andere Teil, die sogenannten Stubs, werden im Verzeichnis /usr/lib abgelegt. Sie werden zu den Programmen dazugelinkt und enthalten nur Verweise auf die eigentlichen Routinen. Ein Update der Libraries kann meist durch Austauschen der Images in /lib vorgenommen werden.

- **/home** - In diesem Verzeichnis wird für jeden Benutzer, außer `root`, ein Home-Verzeichnis eingerichtet. Im jeweiligen Unterverzeichnis werden benutzerspezifische Konfigurationsdateien abgelegt. Außer den persönlichen Dateien der Benutzer sollten hier keine Programme installiert werden. Da dieses Verzeichnis meist auf einer getrennten Partition liegt, ist es nicht ratsam, das Home-Verzeichnis von `root` ebenfalls in diesem Verzeichnis anzulegen. Falls dieses Filesystem aufgrund von Fehlern nicht gemountet werden kann, könnte sich unter Umständen nicht einmal der Administrator in das System einloggen, um den Fehler zu beheben.
- **/install** - Das Installationsprogramm des SLS-Pakets verwendet dieses Unterverzeichnis, um Informationen über installierte Pakete zu speichern.
- **/usr** - Dieses Directory enthält fast alle weiteren wichtigen Unterverzeichnisse, die nicht direkt für das Starten des Systems notwendig sind. Weitere Applikationen werden meist in einem Unterverzeichnis von `/usr` installiert.

Verzeichnisse unter /usr

- **/usr/etc** - Dieses Verzeichnis ist meist ein Verweis auf das Verzeichnis `/etc`. Falls `/usr/etc` als eigenes Verzeichnis existiert, so befinden sich hier nur die Konfigurationsdateien der Programme des `/usr/bin` Verzeichnisses.
- **/usr/bin** - Hier sind die meisten der Systemprogramme bzw. UNIX-Befehle für Benutzer abgelegt. Die Trennung der UNIX-Befehle in solche, die in `/bin` und solche, die in `/usr/bin` abgelegt werden, ist nicht immer konsequent durchgeführt. Im Zweifelsfalle sollte man in beiden Verzeichnissen nachsehen. Sowohl `/bin` als auch `/usr/bin` sollten immer im Suchpfad (PATH) enthalten sein.
- **/usr/bin/X11** - X-Windows-Programme werden normalerweise in diesem Verzeichnis installiert. Es ist jedoch meist nur ein Verweis auf `/usr/X386/bin` und sollte ebenfalls im Suchpfad enthalten sein.

- **/usr/lib** - Wie oben schon erwähnt, werden hier die statischen Libraries für die verschiedenen Programmiersprachen und die Stubs für die Shared Libraries abgelegt. Außerdem enthält dieses Verzeichnis mehrere Unterverzeichnisse, die meist Hilfs- und Konfigurationsdateien anderer Programme beinhalten.
- **/usr/lib/X11** - Hier findet man die Konfigurationsdaten, Zeichensätze, Farbtabellen und andere Dateien des X-Window Systems. Die wichtigste ist dabei `Xconfig`, in der die Konfiguration des X-Servers steht. Dieses Verzeichnis ist meist ein Verweis auf `/usr/X386/lib/X11`.
- **/usr/include** - Hier sind die Include-Files der C-Library abgelegt. Dieses Verzeichnis enhält die Unterverzeichnisse `sys` und `linux`, wobei `linux` nur ein Verweis auf ein Unterverzeichnis von `/usr/src/linux` ist.
- **/usr/man** - Die Manual pages werden in Unterverzeichnissen von `/usr/man` abgelegt.
- **/usr/src** - In den Unterverzeichnissen unter `/usr/src` stehen die Quellcodes der Systemprogramme. Das wichtigste dieser Unterverzeichnisse ist `/usr/src/linux`, in dem sich die Quellcodes des Linux Kernels befinden.
- **/usr/spool** - Unter diesem Verzeichnis werden hauptsächlich Daten zwischengespeichert, die von Daemons empfangen wurden oder weiterverarbeitet werden. Dies sind zum Beispiel die Druckerwarteschlangen oder die empfangenen E-Mail-Nachrichten der einzelnen Benutzer.
- **/usr/adm** - Hier werden meist Protokolldateien des Systems abgelegt.
- **/usr/local** - In diesem Verzeichnis sollten alle zusätzlichen Programme installiert werden, die nicht im Installationspaket enthalten waren. Meist enthält dieses Verzeichnis einen kompletten Unterverzeichnisbaums, bestehend aus einem `bin-`, `lib-`, `etc-`, `include-` und `man`-Verzeichnis. In der Regel wird `/usr/local/bin` in den Suchpfad für Programme und `/usr/local/man` in den Suchpfad für Manual pages aufgenommen.

- **/usr/X386** - Hier beginnt der Verzeichnisbaum des X11-Pakets. Die Verzeichnisse `/usr/lib/X11` und `/usr/bin/X11` sind Verweise in diesen Verzeichnisbaum.
- **/usr/openwin** - In den Unterverzeichnissen von `/usr/openwin` liegen die Programme und Daten des XView-Pakets von Sun. Die Libraries und die Konfigurationsdateien stehen meist im Verzeichnis `/usr/openwin/lib`. Die interessantesten dieser Dateien sind die Definitionsdateien des Menüs für die OpenLook Window-Manager (`olwm` und `olvwm`). Ihre Dateinamen beginnen alle mit `openwin-menue`. Die Windowmanager selbst und die anderen Programme stehen in `/usr/openwin/bin`.
- **/usr/TeX** - In diesem Verzeichnis ist das TeX-Paket installiert.

7.5 Benutzer und Gruppen

Jeder Benutzer besitzt eine eindeutige User Id und gehört zu einer oder mehreren Gruppen. Diese Information ist in den Dateien `/etc/passwd` und `/etc/group` gespeichert.

Um einen Benutzer anzulegen, werden diese Dateien in der Regel nicht mit einem Texteditor geändert. Statt dessen verwendet man das Programm `useradd`.

useradd

Dieses Programm erhält alle wichtigen Angaben über die Kommandozeile und ändert danach die Dateien `/etc/passwd` und `/etc/group` sowie die entsprechenden Shadow-Dateien. Die Shadow-Dateien enthalten die verschlüsselten Benutzer- und Gruppenpassworte in verschlüsselter Form und sind aus Sicherheitsgründen nur vom Superuser lesbar.

Shadow Dateien

Man kann sich die Arbeit wesentlich vereinfachen, indem mit der Option `-D` einmal Standardwerte für die Gruppe, Gültigkeitsdauer des Passworts sowie ein Verzeichnis für die Home-Verzeichnisse der Benutzer vorgegeben werden.

Außerdem können benutzerspezifische Konfigurationsdateien, zum Beispiel `.profile`, `.bashrc` oder `.openwin-menue` im Verzeichnis `/etc/skel` abgelegt werden. Sie werden dann

beim Anlegen eines neuen Benutzers automatisch in dessen Home-Verzeichnis kopiert.
Sind Defaultwerte definiert und die richtigen Dateien in `/etc/skel` abgelegt, so kann man nun einen neuen Benutzer durch Aufruf von

```
linux1:/> useradd -m Benutzername
```

anlegen. Als User Id wird automatisch die nächste freie Nummer verwendet.
Danach muß mit `passwd <Benutzername>` ein Passwort vergeben werden, da der neue Account sonst noch gesperrt ist.
Neben `useradd` gibt es Befehle zum Ändern der Einstellungen eines Benutzers wie `usermod`, `userdel`, `chsh` um die Login-Shell zu ändern oder `chfn` um den vollen Namen zu ändern. Auf diese und weitere Befehle wird in der Manual page zu `useradd` verwiesen.

```
dirk1:/etc# useradd -D -g 6 -b /home -f 3 -e 999
dirk1:/etc# useradd -m peter
dirk1:/etc# passwd peter
Changing password for peter
Enter the new password (minimum of 5 characters)
Please use a combination of upper and lower case letters and
numbers.
New Password:
Re-enter new password:
dirk1:/etc#
```

Festlegen der Defaultwerte und Anlegen eines Benutzers

Gruppen werden mit den Befehlen `groupadd`, `groupmod` und `groupdel` verwaltet. Um einem Benutzer eine zusätzliche Gruppe zuzuweisen kann der Befehl `usermod` mit der Option `-G` verwendet werden.

Anonymous ftp

Eine weitere besondere Rolle neben *root* spielt der Benutzer *ftp*. Falls dieser Benutzer existiert, kann sich ein Anwender mit dem `ftp`-Befehl auf dem Rechner einloggen, ohne daß er einen Account benötigt. Er weist sich einfach als Benutzer *ftp* oder *anonymous* aus, und wird dann vom System aufgefordert, seine

E-Mail-Adresse als Passwort einzugeben. Das Home-Verzeichnis des Benutzers ftp wird dabei als Root-Verzeichnis verwendet, so daß ein Benutzer, der sich auf diese Weise Zugang verschafft, nur auf ganz bestimmte Dateien Zugriff hat.
Dazu muß allerdings im Home-Verzeichnis des Benutzers ftp ein `dev`, `bin` und ein `usr`-Verzeichnis mit entsprechenden Dateien vorhanden sein. Der genaue Aufbau dieses Dateibaums wird in der Manual page zu `ftpd` erklärt.

7.6 Shells

Damit die Benutzer ihre Shell selbständig ändern können, ist zu beachten, daß in der Datei `/etc/shells` jede Shell mit ihrem Zugriffspfad aufgelistet ist. Dies wird häufig vergessen, wenn eine Shell nachträglich installiert wird.

```
/bin/sh
/bin/bash
/bin/ksh
/bin/tcsh
```

Beispiel einer Datei /etc/shells

Ist kein Eintrag für diese Shell in der Datei `/etc/shells` vorhanden, so kann diese nicht von den Benutzern als Login-Shell verwendet werden und man bekommt Probleme mit verschiedenen TCP/IP Programmen.

```
dirk1:/home/stefan# chsh
Changing the login shell for stefan
Enter the new value, or press return for the default

        Login Shell [/bin/sh]: /bin/bash
dirk1:/home/stefan#
```

Aufruf des Kommandos chsh zum Ändern der Login-Shell

Das Kommando `chsh` erlaubt den Benutzern die Änderung der Login-Shell. Auch der Administrator sollte es bei der Konfiguration eines Accounts heranziehen. In diesem Fall muß zusätzlich der entsprechende Benutzername als Parameter übergeben werden.

7.7 Information der Benutzer

Der Text, der vor dem Login-Prompt ausgegeben wird, steht in der Datei `/etc/issue`. In dieser Datei wird in der Regel ein Begrüßungstext mit dem Namen des Rechners und Hinweisen für Benutzer eingetragen.
Hat sich ein Benutzer eingeloggt, so wird meist der Text aus der Datei `/etc/motd` ausgegeben. `Motd` steht dabei für "message of the day". Die Datei sollte auch dementsprechend benutzt werden.

7.8 Backups

Eine relativ einfache Möglichkeit um Backups von wichtigen Dateien anzulegen, bietet der `tar`-Befehl. Er gehört zu den Standard-UNIX-Befehlen, die unter Linux in einer erweiterten Form der FSF vorliegen.

Disketten

Um zum Beispiel alle Daten im Verzeichnis `/home/stefan` auf Disketten zu sichern, kann der `tar`-Befehl mit der Option `M` im Multivolume-Modus aufgerufen werden. Ist die erste Diskette voll wird von `tar` eine weitere angefordert.

```
dirk1:/root# cd /home/stefan
dirk1:/home/stefan# tar cvfM /dev/fd0 *
```

Erstellen eines tar-Archivs auf Disketten

Unterverzeichnisse werden automatisch bei der Archivierung eingeschlossen. Um ein solches Backup wieder auf die Festplatte zu übertragen, muß der `tar`-Befehl wiederum mit der Option M aufgerufen werden, da er sonst nach der ersten Diskette abbricht.

```
dirk1:/root# cd /home/stefan
dirk1:/home/stefan# tar xvfM /dev/fd0
```

Zurückladen eines tar-Archivs von Disketten

Streamer

Genauso können größere Backups auf einem Bandlaufwerk (Streamer) gemacht werden. Dazu muß nur anstelle von /dev/fd0 das Device des Streamers angegeben werden.

7.9 Filesystem Management

Ein weiteres Aufgabengebiet des Systemadministrators ist die Verwaltung der Filesysteme. Im Normalbetrieb beschränkt sich dies darauf, in regelmäßigen Abständen den freien Speicherplatz zu kontrollieren und gelegentlich den Inhalt der `/tmp`-Verzeichnisse zu löschen.

Kommt es jedoch zu einem Systemabsturz, so sollte unbedingt eine Konsistenzprüfung durchgeführt werden. Zu diesem Zweck stellen die einzelnen Linux-Dateisysteme ein spezielles Tool namens `fsck` (File System Check) zur Verfügung. Es ist jedoch darauf zu achten, daß das zu überprüfende Filesystem nicht gemounted ist und auch wirklich das passende Prüfprogramm benutzt wird. Für das momentan meistverwendete `ext2`-Filesystem heißt das entsprechende Tool beispielsweise `e2fsck`.

Filesystem-check

```
linx1:/>umount /dev/hda3
linx1:/>e2fsck /dev/hda3
linx1:/>mount /dev/hda3 /home
```

Konsistenzprüfung eines Dateisystems

Beim Aufruf wird die zu testende Partition bzw. das Filesystem als Parameter übergeben. Normalerweise werden nur eventuelle Inkonsistenzen auf der Konsole ausgegeben. Das Dateisystem ist in Ordnung, wenn keine Fehlermeldung erscheint.

Spezielle Optionen erlauben auch eine automatische Fehlerkorrektur. Meist wird von den Programmautoren aber eine interaktive Reparatur eines defekten Dateisystems empfohlen.

Die Überprüfung eines beim Systemabsturz nicht sauber heruntergefahrenen Dateisystems kann beim `ext2`-System auf Wunsch auch automatisch beim nächsten Systemstart erfolgen. Dazu muß allerdings ein Paket namens `bootutils` installiert worden sein. Bei einigen Linux-Distributionen ist dies standardmäßig der Fall.

bootutils

7.10 Updates

Da gerade bei Linux fast jeden Monat neue Versionenen des Kernels, der C-Library oder des C-Compilers verfügbar sind, ist

eine der Aufgaben des Systemadministrators, diese Updates in das System einzuspielen. Wie man dies macht, wird im folgenden erläutert.

GCC

Die jeweils neueste Version des GNU-C-Compiler (GCC) für Linux kann von dem FTP-Server `tsx-11.mit.edu` geladen werden. Es sind meist mehrere mit `tar` und `gzip` gepackte Dateien, in denen die für Linux compilierten Programme (Binaries) enthalten sind.
Um den alten C-Compiler gegen eine neue Version auszutauschen, reicht es meist aus, die Tar-Dateien im Root-Verzeichnis auszupacken. Die ältere Version wird dabei überschrieben.
Genauere Anweisungen stehen in einem der zugehörigen "Readme" oder "Release" -Dateien, die man zusammen mit den Tar-Archiven findet.

Libraries

Die Linux C-Library ist ebenfalls auf dem FTP-Server `tsx-11.mit.edu` verfügbar. Um eine neue Version der Library zu installieren, benötigt man meist zwei Tar-Archive, deren Namen "`image`" und "`inc`" enthält. In der einen Datei sind die eigentlichen Library-Dateien enthalten, in der anderen die passenden Header-Dateien.
Diese müssen so ausgepackt werden, daß die Include-Dateien in das Verzeichnis `/usr/include` kommen, die Library-Dateien mit der Endung `.sa` und `.a` in `/usr/lib` und die mit der Endung `.so.<version>` in `/lib`.
Die Versionskennung der Libraries besteht aus zwei Teilen, einer sogenannten Major und einer Minor-Version. Die Dateien, die nach `/lib` entpackt werden heißen bei der Major-Version 4 und der Minor-Version 2 beispielsweise `/lib/libc.so.4.2` und `libm.so.4.2`.
Auf diese Dateien wird über einen symbolischen Link zugegriffen, der nur die Major-Version im Namen enthält. Für die

oben genannten Dateien müssen Links mit dem Namen `/lib/libc.so.4` und `/lib/libm.so.4` existieren, die auf die eigentlichen Dateien verweisen.
Wenn eine Library mit einer neuen Minor-Version installiert wird, müssen diese Links verändert werden.

```
linux2:/lib#ls -l libc*
lrwxrwxrwx   1 root     root           13 May 31 17:04 libc.so.4 ->
libc.so.4.3*
-rwxr-xr-x   1 root     other      623620 Apr  1 05:56 libc.so.4.3.3*
-rwxr-xr-x   1 root     other      623620 May 20 12:14 libc.so.4.4*
linux2:/lib#ln -sf libc.so.4.4 libc.so.4
linux2:/lib#ls -l libc*
lrwxrwxrwx   1 root     root           11 Aug 28 13:42 libc.so.4 ->
libc.so.4.4*
-rwxr-xr-x   1 root     other      623620 Apr  1 05:56 libc.so.4.3.3*
-rwxr-xr-x   1 root     other      623620 May 20 12:14 libc.so.4.4*
linux2:/lib#ln -sf libm.so.4.4 libm.so.4
linux2:/lib#
```

Installation einer neuen Library Version

Im obigen Beispiel wird eine Library mit der Major-Version 4 und der Minor-Version 3.3 durch eine neuere Version mit der selben Major-Version und der Minor-Version 4 ersetzt.
Die Option `f` beim Ändern des Links ist notwendig, da die Datei `/lib/libc.so.4` schon existiert und auf die ältere Version verweist.
Es ist nicht ratsam, den alten Link zuerst zu löschen und dann zu versuchen diesen neu anzulegen. Da der Befehl `ln` selbst die C-Library benötigt, kann nach dem Löschen des Links kein neuer Link mehr angelegt werden. Auch alle anderen Befehle sind auf die C-Library angewiesen und können daher nicht mehr aufgerufen werden.
Um den Link wieder einzurichten, muß man mit einer Bootdiskette (z.B. aus dem SLS-Paket) booten, das Root-Filesystem der Festplatte mounten und dort den Link neu erstellen.

Kernel

Die Quelltexte des jeweils neusten Kernels sind auf den meisten Linux FTP-Servern verfügbar. Als erstes erscheint er jedoch auf dem finnischen `nic.funet.fi`. Um ihn zu installieren ist es sinnvoll, zunächst das Verzeichnis `/usr/src/linux` komplett zu löschen und danach das Tar-Archiv mit dem Kernel-Quelltext

im Verzeichnis `/usr/src` auszupacken. Dabei wird das Unterverzeichnis `linux` neu angelegt.
Danach kann man, wie schon im Kapitel über die Konfiguration und Compilation des Kernels beschrieben, mit `make config` die zu verwendenden Treiber neu definieren und dann den Kernel übersetzen.

7.11 Bootdisketten

Falls man das Root-Passwort vergessen, oder bei der Installation einer neuen C-Library einen Fehler gemacht hat und sich nicht mehr einloggen kann, ist eine Bootdiskette meist die einzige Möglichkeit, den Fehler zu korrigieren.
Eine einfache Bootdiskette enthält nur den Linux Kernel, den man mit dem Utility `dd` auf die Diskette geschrieben hat. Das Filesystem, das dieser Kernel als Root-Filesystem mounten soll, kann mit dem Programm `rdev` eingestellt werden.

Falls das Root-Filesystem fehlerhaft ist oder Programme, die zum Booten auf diesem Filesystem benötigt werden, nicht mehr ausführbar sind, so reicht eine normale Bootdiskette nicht aus.
Man benötigt eine Bootdiskette, die gleichzeitig ein eigenes Root-Filesystem enthält. Die SLS-Installationsdiskette A1 ist dazu verwendbar. Nach dem Booten von Diskette, kann man das Root-Filesystem der Festplatte mounten und Korrekturen an den defekten Dateien durchführen.

Support & Hilfe

Ein Argument, mit dem häufig speziell im kommerziellen Bereich Free oder Public Domain Software abgelehnt werden, ist das Fehlen einer Herstellerfirma, die Support und eine Hotline bietet. In den USA gibt es inzwischen Firmen, die dies als Marktlücke erkannt haben, und kommerziellen Support für Free Software anbieten. Auch in Deutschland zeichnet sich mittlerweile eine ähnliche Entwicklung ab.
Unabhängig davon gibt es eine ganze Reihe von Möglicheiten, über das Internet oder direkt aus dem System Information und Hilfe bei konkreten Problemen mit Linux zu bekommen. Dieses Kapitel soll einen Überblick über diese Möglichkeiten und über die verfügbaren Dokumente geben.

8.1 man, xman

Eine einfache Informationsquelle ist, wie bei jedem UNIX-System, die Online-Dokumentation. Dabei handelt es sich um Dateien, die jeweils einen Befehl, eine C-Bibliotheksroutine oder den Inhalt einer Konfigurationsdatei beschreiben. Der Befehl zur Anzeige der sogenannten Manual pages heißt `man`.
Diese Manual pages sind in Abschnitte (sections) von 1 bis 8 eingeteilt, und liegen unterhalb des Verzeichnisses `/usr/man/` in den Unterverzeichnissen `man1` bis `man8` und `cat1` bis `cat8`. Darüberhinaus gibt es oft noch ein Verzeichnis `mann` und `catn`, in denen neue Manual pages liegen.

```
linux2:/usr/man#ls
./      cat1/   cat3/   cat5/   cat7/   catn/   man3/   man6/   mann/
../     cat2/   cat4/   cat6/   cat8/   man1/   man5/   man8/   whatis
linux2:/usr/man#ls man1
./          cpp.1       othello.1   uupath.1    xmag.man    xtetris.1
../         g++.1       pathto.1    uuwho.1     xmake5.1
cccp.1      gcc.1       spider.1    wish.1      xpuzzle.1
linux2:/usr/man#
```

Die Verzeichnisse der Manual pages

Format der Manual pages

Die Manual pages liegen unter Umständen in 2 bis 3 verschiedenen Formaten vor. Das Ausgangsformat sind meist Quelldateien für `nroff`. Sie stehen in den Verzeichnissen `man1` bis `man8`. Diese Darstellung eignet sich nicht zur direkten Bildschirmausgabe, sondern muß mit dem Textformatierprogramm `nroff` bzw. `groff` übersetzt werden. (siehe auch Kap 11.4 Textbearbeitung)
Da dieser Vorgang bei komplizierteren Manual pages einige Zeit dauert, werden die Dateien, nachdem sie einmal umgewandelt worden sind, lesbar in den Verzeichnissen `cat1` bis `cat8` abgelegt. Dazu benötigt der Benutzer allerdings die nötigen Schreibrechte für diese Verzeichnisse. Auf Wunsch können die Dateien auch mit `compress` komprimiert werden, was den Zeitbedarf beim Ansehen nur unwesentlich erhöht, den Platzbedarf jedoch stark verringert.

Falls zusätzliche Manual pages unter einem anderen Verzeichnis als `/usr/man`, zum Beispiel `/usr/local/man` installiert sind, so kann der Suchpfad der Manual pages mit der Environmentvariablen `MANPATH` definiert werden.

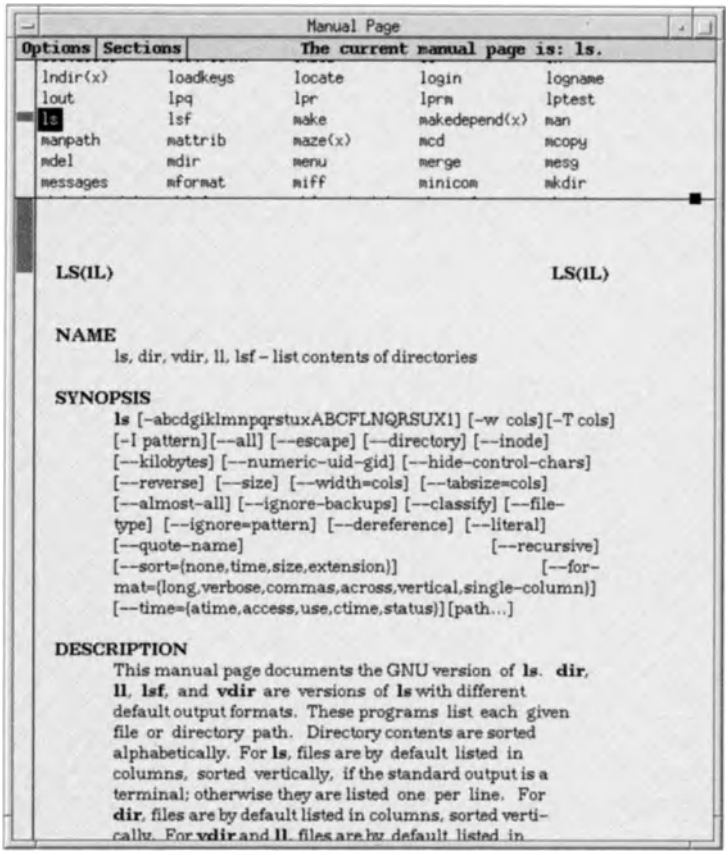
```
setenv MANPATH /usr/man:/usr/openwin/man:/usr/local/man
```

Festlegen des Suchpfades für Manual pages

xman

Für X11 existiert das etwas komfortablere Utility `xman`, bei dem der Benutzer zunächst einen Abschnitt des Manuals auswählen

kann, und dann eine Übersicht aller Manual pages bekommt. Nun kann mit der Maus eine einzelne Manual page ausgewählt werden.



Grafische Ausgabe der Manual pages über xman

Falls man ein Problem mit einem Befehl oder mit der Konfiguration eines Programms hat, so sollte man immer als erstes die entsprechenden Manual pages lesen.

Natürlich existiert auch für das `man`-Kommando eine entsprechende Manual page, die der Benutzer durch Eingabe von `man man` angezeigt bekommt.

8.2 Info

Die Dokumentation vieler Programme, die von der FSF kommen, liegt in einem sogenannten GNU-Info Format vor. Dieses Format wird von GNU-Emacs benutzt, um eine Hypertext ähnliche Navigation durch diese Dokumente anzubieten.
Dazu ruft man innerhalb des Emacs Editors den Info-Modus auf und wählt dann im Menü der verfügbaren Info-Dokumente den gewünschten Text aus. In diesem Text kann man hierarchisch von Stichwort zu Stichwort navigieren. (siehe Kapitel 11 Anwendungen).

```
File: dir       Node: Top       This is the top of the INFO tree
  This (the Directory node) gives a menu of major topics.
  Typing "d" returns here, "q" exits, "?" lists all INFO commands, "h"
  gives a primer for first-timers, "mTexinfo<Return>" visits Texinfo topic,
  etc.
--- PLEASE ADD DOCUMENTATION TO THIS TREE. (See INFO topic first.) ---

* Menu: The list of major topics begins on the next line.

* Info: (info). Documentation browsing system.

* Bison: (bison). bison, and not yacc

* GAWK: (gawk). The GNU version of awk


* Gcc: (gcc). Information about the gcc Compiler

* Cpp: (cpp). The C Preprozessor

* Libg++: (libg++). The G++ Library

--%%-Info:  (dir)Top      (Info Narrow)----Top-----------------------------
Loading info...done
```

Beispiel einer Info page

Neuere Versionen des Emacs Editors, wie Lucid Emacs oder Emacs 19, bieten die Möglichkeit, dies unter der grafischen Oberfläche X11 mit der Maus zu steuern. Es gibt jedoch auch Programme, wie `xinfo` oder das TCL/Tk basierte `tkinfo`, die ausschließlich dazu konzipiert wurden, Info-Dateien anzuzeigen.
Beispiele für Info Dokumente sind die Dokumentation zum GNU C-Compiler, oder zum GNU-AWK Utility.

8.3 Newsgruppen

Da Linux im Internet entwickelt wird, werden selbstverständlich die vielen Kommunikationsdienste des Internet ausgiebig genutzt. Die für Einsteiger interessanteste Informationsquelle sind hier die Newsgruppen.

Für Linux existieren mehrere Newsgruppen, in denen Fragen zum System, der Installation und andere Themen diskutiert werden. Die wichtigsten sind :

- `comp.os.linux.announce` (kurz c.o.l.a)
- `comp.os.linux.help`
- `comp.os.linux.misc`
- `comp.os.linux.admin`
- `comp.os.linux.development`

Bei der Gruppe `comp.os.linux.announce` handelt es sich um eine moderierte Gruppe, in der Ankündigungen neuer Programme oder Portierungen veröffentlicht werden. Nachrichten, die in einer moderierten Gruppe erscheinen sollen, werden nicht direkt, sondern indirekt über einen Moderator eingespielt, der für die Einhaltung der entsprechenden Regeln innerhalb der Gruppe sorgt.
Für die übrigen Gruppen existiert keine Einschränkung. Jeder Teilnehmer kann hier Mitteilungen oder Anfragen veröffentlichen. Man sollte sich jedoch immer bewußt sein, daß eine Nachricht, die in einer Newsgruppe veröffentlicht wird, sich auf der ganzen Welt verbreitet und auf jedem Newsserver Platz benötigt.
Die Anzahl der neuen Nachrichten, die in diesen Newsgruppen täglich erscheinen, liegt derzeit bei über 100. Es ist daher schwierig, ständig auf dem Laufenden zu bleiben, wenn man diese nicht mehrmals pro Woche liest.
Seit einiger Zeit existiert auch eine deutschsprachige Newsgruppe zu Linux (`de.comp.os.linux`). Die dort diskutierten Themen sind jedoch meist nicht so interessant wie die Diskussionen in den englischsprachigen Gruppen.
Oft lassen sich Probleme schon dadurch lösen, daß man regelmäßig die Newsgruppen liest und verfolgt, welche Probleme andere Anwender haben.

8.4 FAQs und HOWTOs

Eine wichtige Informationsquelle, die eng mit den Newsgruppen verbunden ist, sind die sogenannten FAQs. FAQ steht für "frequently asked questions" und ist eine Liste mit häufig gestellten Fragen und entsprechenden Antworten.
FAQs gibt es zu sehr vielen unterschiedlichen Themen. Sie werden meistens von aktiven Teilnehmern einer Newsgruppe verfaßt, um zu vermeiden, daß die gleichen Fragen immer wieder gestellt werden. Der Inhalt einer FAQ ist daher häufig eine Folge solcher Fragen mit den Antworten, die kompetente Leser darauf gegeben haben.
Aus den ursprünglichen Linux-FAQs sind mittlerweile auch sogenannte HOWTOs entstanden. Diese Dokumente sind den FAQs sehr ähnlich, enthalten jedoch einen ausführlicheren Text.

Die Linux-FAQs und HOWTOs enthalten in der Regel keine allgemeinen UNIX-Fragen, da es für derartige Fragen eine eigene Newsgruppe (`comp.unix.questions`) mit einer eigenen FAQ gibt. Für UNIX-Einsteiger lohnt es sich auf jeden Fall auch einen Blick in die UNIX-FAQs zu werfen.

Bezugsquellen für FAQs

Die beste Quelle für FAQs aller Art ist eine Newsgruppe mit dem Namen `news.answers`. Hier werden die FAQs sehr vieler Gruppen regelmäßig abgelegt und neben FAQs zu Linux, UNIX oder Programmiersprachen findet man hier auch so exotische Dinge wie z.B. eine Pyrotechnik-FAQ.
Die speziellen Linux-FAQs werden darüberhinaus regelmäßig in der Newsgruppe `comp.os.linux.announce` abgelegt. Natürlich findet man sie auch auf den vielen FTP-Servern, die meistens ein Verzeichnis `doc` unterhalb von `Linux` haben, in dem die jeweils aktuellen FAQs zu finden sind. Auf dem FTP-Server `nic.funet.fi` liegen diese beispielsweise im Verzeichnis `/pub/OS/Linux/doc/FAQ`.

8.5 Mailing-Listen

Für "Kernel-Hacker" und andere aktive Mitarbeiter des Linux-Systems existieren verschiedene Mailing-Listen, über die vor allem Nachrichten zu neuen Entwicklungen, Problemen, Ideen und Patches ausgetauscht werden.
Nachrichten, die an die Adresse der Mailing-Liste geschickt werden, werden gesammelt und mehrmals täglich an alle Teilnehmer der Liste weitergeleitet.
Um in die Liste zu einem bestimmten Thema, die hier auch *Channel* genannt wird, aufgenommen zu werden, sollte man eine Nachricht an die Adresse `linux-activists-request@niksula.hut.fi` schicken, von der man dann automatisch eine detailierte Anleitung für die Bedienung der Mailing-Liste zurückgesendet bekommt.
Der Befehl dazu lautet im einfachsten Fall :

```
echo help | mail linux-activists-request@niksula.hut.fi
```

Befehl zum Anfordern der Hilfe zu den Mailing-Listen

Da diese Mailing-Listen hauptsächlich für die Entwickler gedacht sind, sollte man als Anfänger keine Fragen in den normalen Channels stellen. Für Anfänger gibt es daher einen speziellen *Newbie* Channel.

8.6 Sonstige Dokumente

Derzeit werden im Internet ausführliche Handbücher zu Linux erstellt. Darunter sind neben einem *Linux Users Guide* und dem *Linux Installation Guide* auch komplexere Dokumente wie der *Network Administration Guide* (kurz NAG), der fast alle Aspekte der Netzwerkinstallation unter Linux abdeckt.
Mit dem Aufbau des Kernels und der Entwicklung von Devicetreibern für Linux beschäftigen sich der *Kernel Hackers Guide* (KHG). Er gibt außerdem einen sehr guten Einblick in die internen Abläufe von Linux.

8.7 Andere Quellen

Neben den oben genannten Quellen gibt es noch viele weitere Internet-Dienste oder Organisationen, über die man Information bzw. Dokumente beziehen kann.

WAIS

WAIS (Wide Area Information System) ist zum Beispiel ein Internet-Service mit dem Dokumente aller Art von beliebigen WAIS-Servern nach Stichworten durchsucht werden können. WAIS bietet sich daher auch für Manuals und FAQs an.
Clients und eine genauere Beschreibung für WAIS findet man zum Beispiel per FTP auf dem Host `think.com` oder über die Newsgruppe `comp.infosystems.wais`.

README-Dateien

Zu den meisten größeren Programmen, aber auch dem Kernel gibt es sogenannte Release Notes und `README`-Dateien, in denen wichtige Hinweise zur Installation und Verwendung stehen. Diese Dateien stehen normalerweise in dem gleichen Verzeichnis wie das System oder der Quellcode des Systems.
So befinden sich zum Beispiel im Verzeichnis `/usr/src` viele Unterverzeichnisse mit Quellcodes zu Systemteilen oder Utilities. In fast jedem dieser Verzeichnisse gibt es eine solche `README`-Datei.
Dementsprechend findet man im Verzeichnis `/etc/lilo` den Linux Loader (LILO) mit einer `README`-Datei, die sowohl Installation als auch Konfiguration im Detail beschreibt.

Free Software Association Germany (FSAG)

Eine ähnliche Organisation wie die Free Software Foundation (FSF) in den USA ist die *Free Software Association Germany* (FSAG). Diese nicht kommerziell orientierte Vereinigung setzt

sich vor allem für die Verhinderung von Software-Patenten und für die Entwicklung freier Software ein.

Sie bietet auch kommerziellen Nutzern von Linux kostenlosen Support via Internet und Telefax. Außerdem wird von der FSAG eine spezielle Version von Linux herausgegeben (Linux/CV), die vor allem kommerziellen Anwendern den Einsatz von Linux erleichtern soll.

Linux/CV

Die FSAG ist inzwischen über folgende EMail-Adresse erreichbar: `fsag@eurom.fsag.rhein-main.de`. Ein eigener HTTP-Server (`http://callisto.fsag.rhein-main.de`) ermöglicht den Zugriff auf die neusten Informationen (siehe Kap 12 Netzwerk-Anwendungen).

X-Windows

9.1 Geschichtliches

Als am Anfang der 80er Jahre die Verbreitung grafikfähiger Workstations begann, gab es kaum Standards für die Programmierung grafischer Benutzeroberflächen. Die meisten Hersteller lieferten zunächst ihre eigenen Oberflächen aus.
Sollte eine Applikation, die von den grafischen Möglichkeiten der neuen Maschinen Gebrauch machte, auf verschiedenen Plattformen lauffähig sein, so mußten große Teile mehrfach entwickelt und gewartet werden. Gerade große Institutionen, die mit Systemen verschiedener Hersteller arbeiteten, bekamen dieses Problem deutlich zu spüren.
Daher entschloß sich das MIT (Massachusetts Institute of Technology) im Rahmen des Athena Projects eine plattformübergreifende, einheitliche Umgebung zur Entwicklung grafischer Applikationen zu entwerfen. Zunächst wurde die Entwicklung des X-Window Systems nur von den Firmen DEC und IBM finanziell unterstützt.

MIT

Im Januar 1987 schlossen sich 12 namhafte Workstation-Hersteller zum X-Consortium zusammen. Ziel dieser Institution sollte es sein, die Weiterentwicklung und Standardisierung von X-Windows voranzutreiben und eine kommerzielle Verwendung zu ermöglichen.
Noch im selben Jahr erschien die Version 11 Release 1. Im Gegensatz zu den früheren Versionen war diese dem Forschungsstadium entwachsen. Zwar war diese neue Version

nicht mehr kompatibel zur Version 10, sie bot aber eine weit größere Flexibilität und Performance, was sie für kommerzielle Zwecke tauglich machte.

9.2 Merkmale

Das X-Windows-System besitzt einige Eigenschaften, die es von herkömmlichen grafischen Oberflächen, wie dem Apple Finder oder MS-Windows unterscheiden. Die nächsten Abschnitte beschreiben die wichtigsten Konzepte dieses leistungsfähigen Systems.

Offenheit

X-Windows wurde, im Gegensatz zu den meisten anderen grafischen Oberflächen, von Anfang an als offenes System konzipiert. Das bedeutet, daß keinerlei herstellerspezifische Politik betrieben wird und die kompletten Quelltexte jedermann zur Verfügung stehen.
X-Windows erlaubt die komfortable Entwicklung portabler und hardwareunabhängiger Software. Der Programmierer muß sich nicht um die zugrunde liegende Hardware kümmern. Es werden eine Vielzahl von Ein- und Ausgabegeräten unterstützt. Außerdem sind auch Schnittstellen für herstellerspezifische Erweiterungen vorgesehen. Dadurch wird die Einbindung von Spezialhardware ermöglicht.
Die Herstellerunabhängigkeit und nicht zuletzt die sehr hohe Portabilität von X-Windows sorgten für eine hohe Akzeptanz im Workstationbereich. Heute gibt es wohl keine Plattform mehr, für die das System nicht verfügbar wäre, egal ob Mainframe oder PC. Von diesem Umstand profitieren natürlich auch die Anwender von Linux. Der unter Linux gebräuchliche X-Window Server X386 zeichnet sich durch eine besonders hohe Performance auf normaler PC-Hardware aus und ist kommerziellen X-Servern durchaus ebenbürtig, in einigen Bereichen sogar überlegen.

Client/Server-Architektur

Das X-Window System unterscheidet aufgrund seiner internen Struktur zwischen dem sogenannten X-Server und den X-Clients. Das Server-Programm ist für die Verwaltung der lokalen Hardware wie Bildschirm, Tastatur und Maus verantwortlich und stellt das Bindeglied zwischen dem Benutzer und den einzelnen X-Applikationen, den sogenannten X-Clients, dar.
Auf jeder Workstation läuft im allgemeinen immer nur ein Server, der beliebig viele Clients mit Eingaben versorgt bzw. die vom Client gewünschten Bildschirmausgaben durchführt.
Es ist jedoch durchaus möglich, daß ein Server mehrere Bildschirme, die an einer Arbeitsstation angeschlossen sind, verwaltet, was vor allem bei CAD-Systemen zur Anwendung kommt.

Die einzige Verbindung zwischen dem Server und den X-Clients stellt das sogenannte X-Protokoll dar. Aufgrund dieses standardisierten Protokolls können die Clients auch auf anderen Rechnern im Netz laufen (Netzwerktransparenz).

X-Protokoll

Man sollte sich bei den Bezeichnungen X-Server und X-Client die Unterschiede zur gewohnten Terminologie klar machen. Normalerweise versteht man unter einem Server eine den Clients hardwaremäßig überlegene Maschine, die die Datenanfragen oder Rechenaufgaben der Clients bearbeitet.
Unter dem X-Window System ist dies meist gerade umgekehrt: der Server läuft auf einer normalen Arbeitsplatzstation, während der Client auf einer leistungsfähigeren Maschine arbeitet. Oftmals ist diese Trennung jedoch nicht vorhanden, da sich Server und Client auf der selben Maschine befinden.

Diese Unterscheidung zwischen Client und Server führt zu einigen Konsequenzen sowohl für den Programmierer als auch für den Anwender.
So muß zum Beispiel der Programmierer eine Grafik, die im Client als Bitmap vorliegt, erst über das Netzwerk zum Server übertragen, ehe diese auf dem Display dargestellt werden kann.

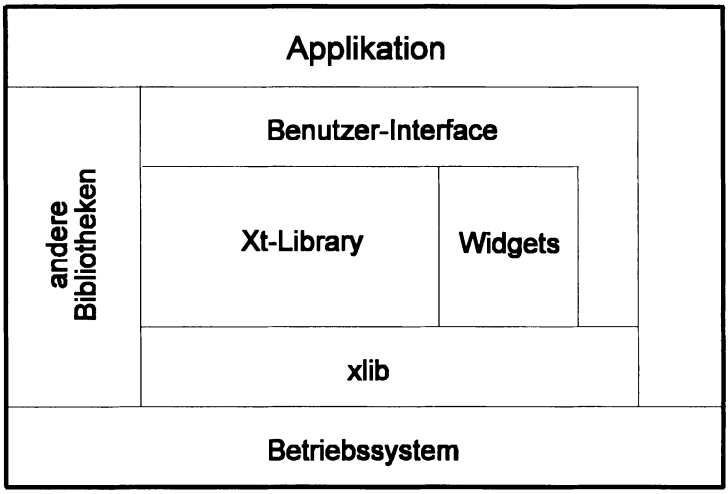
Bei der Entwicklung von X-Applikationen müssen diese Zusammenhänge beachtet werden, um nicht unnötige Netzwerkbelastungen und die damit verbundenen Geschwindigkeitseinbußen zu provozieren.
Da ein X-Client mit seinem jeweiligen Server nur relativ lose gekoppelt ist, ergeben sich für den unerfahrenen Anwender gelegentlich einige Überraschungen. Reagiert ein Client beispielsweise aufgrund hoher Netzwerk- oder CPU-Belastungen nicht sofort auf eine Benutzereingabe, so neigen viele Benutzer dazu, die Eingaben zu wiederholen. Da der X-Server jedoch jedes Ereignis aufzeichnet und dies definitiv dem Client schickt, wird die Aktion eventuell mehrfach ausgeführt, was in den seltensten Fällen dem Wunsch des Benutzers entspricht.

Netzwerktransparenz

Ein wichtiges Merkmal des Systems ist die sogenannte Netzwerktransparenz. Die meisten Workstations im UNIX-Bereich sind in einem Netzwerk miteinander verbunden. X-Windows ermöglicht es nun, die grafischen Ausgaben einer Applikation auf eine beliebige Workstation im Netz umzulenken.
Dieses Feature erlaubt es beispielsweise, rechenintensive Programme auf der jeweils leistungsstärksten CPU laufen zu lassen und die Ein- und Ausgaben auf einer kleineren Arbeitsstation im Netz durchzuführen.
Steht eine entsprechend schnelle Netzwerkverbindung zur Verfügung, können die beiden Maschinen durchaus auch tausende von Kilometern voneinander entfernt sein.
X-Windows ist prinzipiell nicht an ein bestimmtes Netzwerkprotokoll gebunden, obwohl es momentan nur TCP/IP und DECnet unterstützt.
Erstaunlicherweise führt die Netzwerktransparenz im Vergleich zu herkömmlichen Grafiksystemen kaum zu Geschwindgkeitseinbußen. Da es darüber hinaus möglich ist, lokale Applikationen und Programme von verschiedenen im Netz befindlichen Maschinen auf einem Display darzustellen, stellt das X-Window System eine ideale Integrationsbasis für alle möglichen Anwendungsbereiche dar.

9.3 Aufbau

Der Aufbau des X-Window Systems kann in mehrere Schichten aufgeteilt werden. Der Anwendern kommt mit dieser Schichtung im allgemeinen nicht in Berührung. Dennoch dürfte ihre Kenntnis zur Klärung einiger Phänomene beitragen, die auch im Alltag eines Anwenders auftreten.



Aufbau einer X-Window Applikation

Xlib und Protokoll

Die Basis des X-Window Systems stellt eine äußerst umfangreiche Grafikbibliothek mit einer standardisierten C-Schnittstelle names Xlib dar. Jeder Aufruf einer Xlib-Routine wird in einen entsprechend kodierten Datenstrom umgesetzt, der dann über ein Netzwerk übertragen (X-Protokollebene) und auf einer anderen Workstation interpretiert werden kann. Dies garantiert auch die problemlose Kommunikation zwischen Workstations unterschiedlicher Hersteller.
X-Windows erlaubt keinerlei direkte Zugriffe auf die Videohardware und stellt auch keine Möglichkeit zur Verfügung, diese standardisierte Schnittstelle zu umgehen.

Intrinsics

Da Xlib nur rudimentäre Grafikoperationen wie z.B. das Zeichnen von Linien und Kreisen oder das Füllen von Flächen erlaubt, wurden höhere Schichten eingeführt. Das MIT hat mit dem sogenannten X-Toolkit einen ausgereiften Vorschlag zur Implementierung solcher Bibliotheken gemacht. Für die Gestaltung von grafischen Oberflächen werden meist Objekte auf der Toolkit-Ebene benutzt. Werden in einer Anwendung jedoch zusätzlich primitive grafische Ausgabeoperationen benötigt, so werden diese über direkte Xlib-Aufrufe realisiert.

Xt-Library

Die X-Toolkit-Intrinsic-Library erlaubt die Entwicklung und Verwendung komplexer grafischer Objekte wie Buttons, Texteingabefelder oder Auswahlmenüs. Diese Objekte werden im allgemeinen Widgets genannt.

Widgets

Das tatsächliche Aussehen und Verhalten (Look and Feel) solcher Objekte wird durch die Intrinsics-Bibliothek jedoch nicht näher festgelegt. Es war ein Design-Ziel von X, das Erscheinungsbild der darauf aufbauenden Applikationen nicht vorzuschreiben, sondern nur definierte Schnittstellen zur Implementierung und Anwendung solcher Widget-Sets zur Verfügung zu stellen. Daher wurden im Laufe der Zeit von verschiedener Seite derartige Widget-Bibliotheken mit zum Teil erheblichen Unterschieden in Aussehen und Bedienung geschaffen. Den Anwender dürfte diese Vielfalt jedoch eher verwirrt haben. Jüngste Entwicklungen zeigen jedoch, daß sich die meisten Hersteller auf das Motif-Widget-Set der Open Software Foundation als de fakto Standard geeinigt haben und somit in Zukunft wahrscheinlich ein einheitliches Konzept für grafische Benutzeroberflächen unter X-Windows zur Verfügung stehen wird.

9.4 X-Resources

Die Entwickler des X-Window Systems haben beim Design eine ungewöhnlich flexibel gestaltete Schnittstelle zur Konfiguration

verschiedener Parameter des Systems geschaffen. Grundlage dafür ist der objektorientierte Ansatz des X-Toolkits bzw. der darauf aufbauenden Widget-Sets.

Widgetattribute

Jedes dieser Widgets verfügt über eine Anzahl bestimmter Attribute, wie Position, Größe, Form oder Farbe. Diese Merkmale können sowohl vom Programmierer als auch später vom Anwender beeinflußt werden.

Jedes grafische Objekt besitzt zunächst eine interne Standard-Einstellung, die von den Widget-Entwicklern festgelegt wurde. Der Programmierer einer X-Applikation ändert diese im allgemeinen nur in dem Maße, wie es für die jeweiligen Bedürfnisse notwendig ist.

Beim Start einer X-Applikation lädt der Resource-Manager eventuell neue Daten aus der X-Resource-Database, deren Inhalt vom Anwender in mehreren Stufen modifiziert werden kann. Zunächst wird zu den meisten X-Clients eine ensprechende Application-Default-Datei geliefert, die in den X-Windows Systembereich kopiert wird. Diese enthält für die Anwendung wichtige Basiseinstellungen, wie Größe und Position der benutzten grafischen Objekte oder Fehlermeldungstexte in der jeweiligen Landessprache. Darüberhinaus kann sich jeder Anwender in seinem Home-Verzeichnis eine eigene sogenannte `.Xdefaults`-Datei erzeugen und die Einstellungen nach Belieben überlagern.

Die Repräsentation dieser Resource-Werte findet im ASCII-Format statt und kann sich für den unerfahrenen Anwender recht kompliziert darstellen. Zur Unterscheidung innerhalb einer Resource Database bekommt jede Applikation von ihrem Programmierer einen Namen (Klasse) zugeordnet, der nicht unbedingt dem Namen der Programmdatei (Instanz) entsprechen muß. Auch jedes Widget, das von außen konfigurierbar ist, besitzt eine derartige Bezeichnung. Um ein Widget eindeutig referenzieren zu können, genügt die Angabe des Namens jedoch nicht. Es muß vielmehr, ähnlich wie bei einem Dateisystem, ein Pfad angegeben werden, der die Hierarchie der einzelnen Widgets repräsentiert. Um die Attribute mehrerer Widgets gleichzeitig manipulieren zu können, sind innerhalb eines solchen Pfades auch Wildcards erlaubt.

```
XTerm*ScrollBar:            true
XTerm*Foreground:           white
XTerm*Background:           gray20
XTerm*IconName:             XTerm
XTerm*WaitForMap:           true
XTerm*MarginBell:           false
XTerm*JumpScroll:           true

Editor.view.background:     gray
Editor.close.labelString:   Close
Editor.open.labelString:    Open
Editor*background:          gray90
```

Ausschnitt aus einer Resource Database

Die erste Spalte der Resource-Datei spezifiziert jeweils das zu manipulierende Attribut. Dies entspricht meistens einer Widget-Resource. Der Programmierer kann jedoch auch neue, applikationsspezifische Ressourcen, definieren. Die Hierarchie und den Namen der vorhandenen Ressourcen eines Programms kann man meist der zugehörigen Manual page entnehmen.
Release 5 des X11-Systems enthält einen interaktiven Resource-Manager (`editres`), der die komfortable Manipulation aller Resource-Werte eines laufenen Programms erlaubt und diese auf Wunsch in eine ASCII-Datei sichert. Bemerkenswert ist, daß dies zur Laufzeit eines Programms möglich ist. Auf diese Weise kann sich der Benutzer sofort ein Bild über die Auswirkungen seiner Änderungen machen. Leider wird das für `editres` benötigte Protokoll noch nicht von allen Widget-Sets unterstützt, was die universelle Einsetzbarkeit dieses Tools natürlich erheblich einschränkt

9.5 Window-Manager

Einen besonderen Client stellt der sogenannte Window-Manager dar. Auch er existiert meist nur einmal pro Arbeitsplatz und hat die Aufgabe, die unterschiedlichen Fenster eines Displays zu verwalten. Er erlaubt die Beeinflussung von Position und Größe eines Fensters durch den Benutzer. Außerdem ist er für die sogenannte Fensterdekoration zuständig.
Als Dekoration wird der Randbereich eines Fensters mit seinen verschiedenen Bedienungselementen bezeichnet. Für den Fenster-

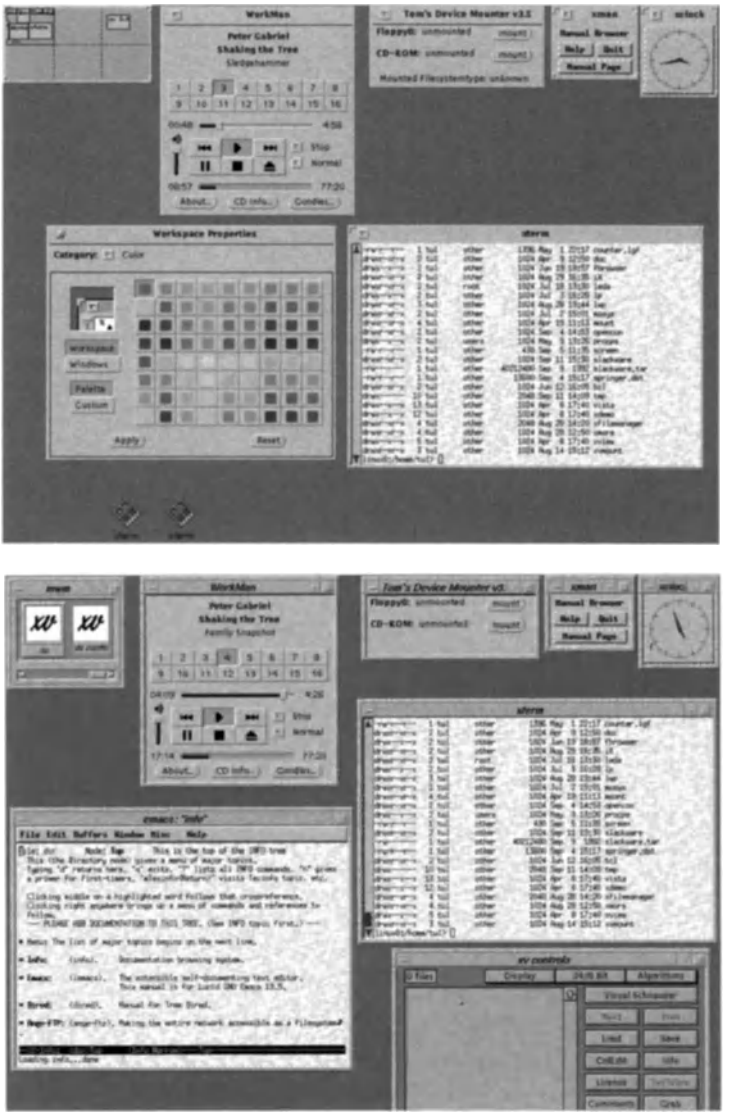
inhalt ist selbstverständlich der entsprechende X-Client verantwortlich.

Auch für den Window-Manager gilt, daß weder das Erscheinungsbild noch die Art der Bedienung vom X-Window System vorgeschrieben wird.

Um dennoch die problemlose Kommunikation zwischen dem jeweiligen Window-Manager und den unter seiner Kontrolle ablaufenden Clients zu gewährleisten, wurden vom X-Consortium die sogenannten ICCC (Inter Client Communications Conventions) verabschiedet.

Dem Anwender stehen inzwischen viele unterschiedliche Window-Manager zur Verfügung, die jeweils ihre Vor- und Nachteile besitzen. Die Standard X-Distribution enthält beispielsweise den `twm` (Tab Window Manager), der jedoch keinen hohen Bedienungskomfort bietet.

Von der Open Software Foundation (OSF) stammt der Motif-Window-Manager (`mwm`), der von seinem Erscheinungsbild zum zugehörigen Motif-Widget-Set paßt. Der von Sun Microsystems entwickelte OpenLook-Window-Manager (`olwm`) steht im Quelltext kostenlos zur Verfügung und wurde von dritter Seite zum sogenannten OpenLook Virtual-Window-Manager (`olvwm`) erweitert. Er stellt nahezu beliebig viele virtuelle Bildschirme zur Verfügung, zwischen denen über den Desktop-Manager umgeschaltet werden kann.



Olvwm und Mwm im Vergleich

Die freie Verfügbarkeit und die Erweiterung des Platzangebotes auf dem Bildschirm machen den `olvwm` zum momentan idealen Window-Manager, nicht nur für Linux.

Darüber hinaus gibt es den Generic Window-Manager (`gwm`), dessen Look and Feel vom Benutzer an die beiden verschiedenen Standards (OpenLook und Motif) angepaßt werden kann.

Die Window-Manager `ctwm` und `fvwm` basieren beide auf dem `twm`, bieten jedoch ein ansprechenderes Erscheinungsbild und sind besser konfigurierbar. Für Systeme mit sehr wenig Hauptspeicher eignet sich besonders der `fvwm`.

9.6 Toolkits

Der nachfolgende Abschnitt stellt die wichtigsten Bibliotheken (Toolkits) zur Erstellung grafischer Benutzeroberflächen unter X-Windows vor.

Athena-Widget-Set

Das Standard-Widget-Set des X-Window Sytems ist das sogenannte Athena-Widget-Set. Es war das erste verfügbare Widget-Set für X und liegt der MIT-Distribution bei. Viele der frühen X-Applikationen benutzten die Athena-Widgets. Auch heute werden diese noch von vielen Freeware Programmen benutzt.
Leider wirkt die optische Erscheinung und die Art der Bedienung einiger Elemente etwas antiquiert, so daß die meisten kommerziellen Anwendungen inzwischen an modernere Toolkits angepaßt wurden. Dies war aufgrund der Ähnlichkeiten zwischen Intrinsic basierten Toolkits wohl kein größeres Problem.
Doch auch die Form und Art der Darstellung der Athena-Widgets hat in jüngster Zeit eine Aufbesserung erfahren. Da auch diese Quelltexte frei verfügbar sind, war es möglich, die Zeichenroutinen auszutauschen. Die durchgeführten Modifikationen führten zu einem Motif-ähnlichen 3D-Erscheinungsbild. Interessanterweise kann bei Linux die alte shared Library (`libXaw`) einfach gegen die neue (`libXaw3D`) ausgetauscht werden.

OpenLook

OpenLook stellt zunächst nur eine detailierte Spezifikation des Look and Feels für grafische Benutzeroberflächen dar und wurde

nach mehrjähriger Entwicklung von AT&T und Sun Microsystems verabschiedet. Ziel war es, einen Standard für Benutzerschnittstellen unter X-Windows zu schaffen.

XView

Sun entwickelte auf der Basis der OpenLook-Spezifikation ein zu ihrem bisherigen Grafiksystem SunView nahezu kompatibles Toolkit für X-Windows (XView). Dieses wurde der Relase 4 der MIT-Distribution von X11 beigelegt und ist somit frei verfügbar. Sun benutzte dieses Toolkit, das leider direkt auf Xlib aufsetzt und somit kein Widget-Set im eigentlichen Sinne darstellt, um eine Reihe von Standardanwendungen zu implementieren, die mit jeder Sun Workstation ausgeliefert werden (OpenWindows Deskset).

AT&T versuchte durch die Auslieferung dieser Oberfläche mit den UNIX System V Systemen für eine weite Verbreitung zu sorgen. Als Alternative zu XView wurde, in erster Linie von AT&T, ein X-Toolkit basierendes Widget-Set entwickelt, das ebenfalls den OpenLook-Spezifikationen entspricht.

OSF/Motif

Um die Entwicklungen neuer Technologien und Standards vor allem für UNIX-Systeme voranzutreiben, schlossen sich die meisten namhaften UNIX-Hersteller (außer Sun und AT&T) zur OSF (Open Software Foundation) zusammen. Darunter finden sich Firmen wie Hewlett Packard, IBM und DEC.

Auch diese Gruppe erkannte den Mangel an einer einheitlichen Benutzerschnittstelle für X-Windows und gab die Entwicklung von OSF/Motif in Auftrag. Vor allem die Firmen DEC und HP waren maßgeblich am Design und der Implementierung beteiligt. Innherhalb weniger Monate entstand ein am Erscheinungsbild gängiger Oberflächen im PC-Bereich angepaßtes User-Interface. Hauptkomponenten waren der Motif Window-Manger (mwm) und das auf den X-Intrinsics basierende Motif-Widget-Set.

UIL

Außerdem wurde eine formale Sprache geschaffen, die die einfache Beschreibung Motif-basierter grafischer Benutzeroberflächen ermöglicht, die sogenannte User Interface Language (UIL). Diese Beschreibung kann mit Hilfe eines speziellen

Compilers in ein binäres Format überführt und mit Hilfe einer Bibliothek interpretiert werden.
Von zahlreichen Drittanbietern stehen inzwischen Programme zur interaktiven Erstellung von Motif-Oberflächen zur Verfügung.

Aufgrund der großen Anzahl der an der OSF beteiligten Firmen entwickelte sich Motif nach einiger Zeit zum de fakto Standard in der UNIX-Welt.
Die OpenLook-Entwickler Sun und AT&T (heute UNIX System Laboratories bzw. Novell) schlossen sich im Frühjahr 1993 in der COSE-Initiative dem Motif-Interface an. Somit dürfte OpenLook, zumindest was den kommerziellen Bereich angeht, wohl der Vergangenheit angehören.

Da die OSF den Quelltext von Motif nicht kostenlos verfügbar machte, müssen für jede Laufzeit-Lizenz Gebühren abgeführt werden. Aus diesem Grund stand Motif auch nicht von Anfang an für Linux zur Verfügung. Mittlerweile sind jedoch kommerzielle Linux-Versionen von OSF/Motif zu einem günstigen Preis lieferbar.
Hochschulen können die Quelltexte direkt bei der OSF erwerben und auf ihren Workstations übersetzen. Dies wurde auch von verschiedenster Seite auf Linux-Systemen durchgeführt.

SUIT

Suit (Simple User Interface Toolkit) wurde an der University of Virginia entwickelt. Im Gegensatz zu den meisten X-Toolkits steht dieses auch unter anderen gängigen grafischen Benutzeroberflächen, wie MS-Windows oder dem Apple Macintosh, zur Verfügung.
Mit Hilfe von SUIT ist es möglich, zwischen diesen Systemen portable Applikationen zu entwerfen. Das Look and Feel ist stark an OSF/Motif angelehnt, was die Akzeptanz beim Benutzer erhöhen dürfte. Ein besonderes Merkmal dieser Bibliothek ist der integrierte Interface-Editor. Dem Benutzer wird damit die einfache interaktive Manipulation der Oberfläche ermöglicht. Er kann die Objekte beispielsweise in Position und Form

modifizieren, ohne daß eine Neuübersetzung notwendig wäre. Alle Änderungen werden beim Verlassen der Applikation automatisch in eine Datei gesichert.
Nachteilig ist jedoch, daß der jedem Programm hinzugelinkte Code des Editors die Größe erheblich erhöht. Mittlerweile ist jedoch schon eine Shared Library Version erhältlich, die dieses Problem entschärft. Unangenehm fällt auch die relativ träge Reaktion auf Benutzereingaben auf.

InterViews

Bei InterViews handelt es sich um eine umfangreiche grafische Umgebung, die an der Stanford University entworfen wurde. Im Gegensatz zum X-Toolkit bzw. den zughörigen Widget-Sets verfügt InterViews über eine C++-Schnittstelle und folgt damit objektorientierten Konzepten. Das InterViews-System stellt neben einer Grafikbibliothek einen interaktiven Interface-Editor (`ibuild`), einen Texteditor mit WYSIWYG-Darstellung (`doc`), einen C++ Class-Browser (`iclass`) und ein leistungsfähiges, vektororientiertes Grafikprogramm (`idraw`) zur Verfügung.
Aufgrund des recht großen Overheads der meisten objektorientierten Systeme stellt auch InterViews nicht unerhebliche Anforderungen an die Hardware.
Die Bedeutung von InterViews wird jedoch mit der kommenden Release 6 von X11 wachsen. Die neue Widget-Technologie namens *Fresco* basiert in wesentlichen Teilen auf den Konzepten von InterViews.

Tcl/Tk

Tcl/Tk (sprich tickl) stellt ebenfalls eine Universitäts-Entwicklung (Berkeley) dar und erlaubt die Erstellung von Scripts mit grafischen Frontends. Tcl (Tool Command Language) alleine hat mit grafischen Benutzeroberflächen zunächst nichts zu tun. Es handelt sich dabei vielmehr um eine Interpretersprache, die als C-Library vorliegt. Daher läßt sie sich recht einfach in bestehende C-Programme einbinden. Tcl besitzt sehr mächtige

Konstrukte zur Verarbeitung von Listen und assoziativen Arrays und erlaubt die Erweiterung des Sprachumfangs über eine C-Schnittstelle.

Tk ist ein Widget-Set, das neben einer C-Schnittstelle über eine Tcl-Schnittstelle verfügt. Es eignet sich daher besonders dazu, Tcl-Programme mit einer grafischen Oberfläche zu versehen. Wegen der zugrundeliegenden Interpretersprache lassen sich sehr einfach und schnell Prototypen aber auch komplette Applikationen entwickeln.

Für Tcl/Tk gibt es ein Programm mit Namen XF, mit dem eine Tk-Oberfläche grafisch und interaktiv erstellt werden kann. Damit können Tk-Oberflächen nicht nur einfach realisiert werden, sondern man kann auch bestehende Tcl/Tk-Programme laden und nachträglich modifizieren oder erweitern. Da XF selbst in Tcl/Tk entwickelt wurde, kann die Oberfläche bereits im Verlauf der Entwicklung getestet werden und die Darstellung während der Entwicklung entspricht exakt der des fertigen Programms.
Neben Tcl/Tk gibt es auch Tcl/Motif, das den Zugriff auf OSF/Motif-Widgets von Tcl aus ermöglicht.
Auf einige Details von Tcl wird später noch im Kapitel 10.5 eingegangen.

XView/Slingshot/UIT

XView ist ein unabhängiges Toolkit, das nicht auf den Xt-Intrinsics aufsetzt. Es stammt von der Firma Sun und ist mit der Release 4 des X-Window Systems ausgeliefert worden. Die Programmierschnittstelle ist stark an das inzwischen veraltete SunView angelehnt. SunView war eine nicht netzwerktransparente, eng mit dem Kernel verbundene grafische Oberfläche der frühen Sun Workstations. XView implementiert das bereits erwähnte Look and Feel von OpenLook und ist, aufgrund des objektorientierten Ansatzes und der nicht allzu komplizierten Struktur, relativ einfach zu programmieren.

Um die Möglichkeiten von XView auszubauen, wurden von einem Mitarbeiter von Sun England die sogenannten Slingshot-Extensions entwickelt. Dieses Paket bietet zahlreiche neue grafische Objekte, die sich harmonisch in das Basissystem eingliedern. Slingshot erweitert den objektorientierten Ansatz um rudimentäre grafische Figuren wie Linien oder Rechtecke. Auch die Programmierung der in OpenLook relativ ausgereiften Drag-und-Drop Funktionalität wird durch die Slingshot-Extensions vereinfacht.
Um XView (und Slingshot) an die neuen Entwicklungen im Bereich der Programmiersprachen anzupassen, wurde, ebenfalls von einem Sun Mitarbeiter, eine C++ Klassenhierarchie (UIC) entworfen. Diese erlaubt die Benutzung aller XView-Objekte und vereinfacht den Umgang mit diesen. Leider ist die Integration der Funktionen in abgeleitete C++-Klassen nicht ganz unproblematisch.

9.7 Interfacebuilder

Die komfortabelste Möglichkeit, X-Window basierte Applikationen zu erstellen, bieten spezielle Programme zur interaktiven Gestaltung grafischer Oberflächen. Diese Interfacebuilder sind meistens in der Lage, C-Quelltext zu erzeugen. Dieser kann dann vom Programmierer beliebig erweitert werden.

ParcPlace, ObjectBuilder

Mittlerweile sind auch für Linux einige Tools zum interaktiven Design grafischer Oberflächen erhältlich. Die Firma *ParcPlace* bietet mit dem *ObjectBuilder* ein äußerst leistungsfähiges Werkzeug zur Oberflächen-Entwicklung in C++ an. Die zugehörige Bibliothek *ObjectInterface* stellt eine Xlib-basierte C++-Klassen-Hierarchie zur Verfügung, deren Look and Feel zur Laufzeit zwischen OpenLook und Motif umgeschaltet werden kann.

9.8 Der X386 Server

Der X-Server unter Linux entstammt dem XFree386 genannten X-Windows-Paket für PC-UNIX Implementierungen. Er wurde

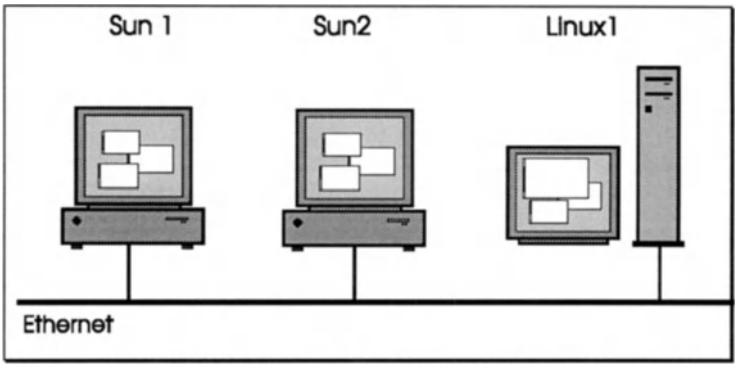
speziell an die im PC-Bereich üblichen Grafikkarten angepaßt und besitzt eine beachtliche Performance, die teilweise über das hinausgeht, was man von einer RISC-Workstation erwarten kann. In der Standard-Konfiguration werden nur die gängigen VGA-Grafikkarten in allen möglichen Auflösungen und Bildwiederholfrequenzen unterstützt. Mit speziellen Servern, die die Möglichkeiten moderner Beschleunigerkarten ausnutzen, lassen sich jedoch erheblich bessere Ergebnisse erzielen.

9.9 Praxis

Der folgende Abschnitt zeigt zunächst, wie sich der Anwender auf einer anderen Workstation einloggt, um die Ausgaben von dort laufenden X-Windows-Programmen auf seine lokale Linux-Maschine umzulenken. Anschließend wird kurz auf die Konfiguration des X-Window Systems aus Anwendersicht und die Möglichkeit einer speicherplatzoptimierten X-Windows-Umgebung eingegangen.

Linux als X-Terminal

Ein unter Linux und XFree laufender PC eignet sich hervorragend als X-Windows-Terminal. Dies ist vor allem dann interessant, wenn teure Softwarepakete auf einer Workstation zur Verfügung stehen, die dezentral genutzt werden sollen.
Mittlerweile sind auch zahlreiche X-Server erhältlich, die unter MS-Windows laufen. Deren Performance ist jedoch meist erheblich geringer als die des XFree 386 Servers, da die X-Grafikroutinen auf die langsameren MS-Windows-Aufrufe umgesetzt werden.
Um sich von diesen Möglichkeiten ein genaueres Bild machen zu können, soll im folgenden ein Beispiel für eine derartige Anwendung gezeigt werden.



Heterogenes Netzwerk

Auf einer Sun Workstation (`sun1`) in einem Netzwerk steht beispielsweise ein leistungsfähiges Grafikprogramm zur Verfügung, das von einem Linux-Arbeitsplatz (`linux1`) benutzt werden soll. Der Linux-PC muß dazu über Ethernet mit der Workstation verbunden sein. Zunächst muß der externe Zugriff auf den eigenen X-Server zugelassen werden. Dazu dient das Kommando `xhost`. Die Zugriffsberechtigung kann für jede Maschine getrennt definiert werden. Im allgemeinen genügt jedoch die Ausführung von `xhost +`.

```
linux1:/home/stefan>xhost +
access control disabled, clients can connect from any host
xhost:  must be on local machine to enable or disable access
control.
linux1:/home/stefan>
```

Anwendung des `xhost`-Befehls

Anschließend loggt sich der Linux-Anwender über `telnet` oder `rlogin` auf dem Rechner `sun1` ein.

```
linux1:/home/stefan>
linux1:/home/stefan>telnet sun1
Trying 141.7.1.20...
Connected to sun1.
Escape character is '^]'.


SunOS UNIX (sun1)

login: strobel
Password:
Last login: Mon Aug 30 09:59:51 from SunServ
SunOS Release 4.1.2 (NEWKERN) #1: Wed Dec 23 11:02:57 MET 1992
sun1:/home2/sun1/strobel>
sun1:/home2/sun1/strobel> setenv DISPLAY linux1:0.0
```

Einloggen auf einem anderen Rechner mit `telnet`

Nun wird die sogenannte DISPLAY-Environment-Variable gesetzt, damit die X-Library alle grafischen Ausgaben auf die Linux-Maschine umlenkt. Dies geschieht bei einer C-Shell, wie im obigen Beispiel gezeigt, mit dem Befehl `setenv`.
Wird nun auf `sun1` ein X-Client gestartet, so erscheinen alle Fenster nicht auf `sun1`, sondern auf dem Display des Linux-Rechners mit dem Namen `linux1`.
Auf diese Weise können auch mehrere Applikationen von verschiedenen Workstations auf ein Display ausgegeben und von einer Maschine aus bedient werden.
Um die Umlenkung einer Remote-Applikation zu vereinfachen, bietet es sich unter Linux an, einen entsprechenden Eintrag im Workspace-Menü des `olvwm` Window-Managers vorzusehen. Dies geschieht in der Datei `.openwin-menue`.
In diesem Fall wird die Applikation mit Hilfe einer Remote-Shell (`rsh`) von Linux aus auf einer anderen Maschine im Netz gestartet.

```
#  openwin-menu
#  OpenWindows root menu file - top level menu

Remote MENU
    "Sun1..."   exec rsh -n linux1 "xterm -display linux1:0.0"
Remote END PIN
```

Remote-Menü in einer `.openwin-menue`-Datei

Die im Beispiel markierte Befehlssequenz läßt sich nun unter der Bezeichnung "Sun1..." im Workspace-Menü auswählen.
Die Option `-display` erlaubt die Angabe der Station, auf der die Ausgaben erscheinen sollen. Im vorliegenden Beispiel wird also auf `sun1` über `rsh` ein `xterm` Emulator gestartet, dessen Fenster auf `linux1` erscheint.

Konfiguration

Je nach Window-Manager und Konfiguration des X-Window Systems hat der Benutzer einige Möglichkeiten, sich seine individuelle Umgebung zu schaffen.

Die wichtigste Datei, für benutzerspezifische Einstellungen innerhalb einer X-Window Umgebung ist `.Xdefaults`. Sie liegt im Home-Verzeichnis des Benutzers.

```
*fontList:  fixed

xterm*ScrollBar:        true
xterm*Foreground:       white
xterm*Background:       grey20
xterm*IconName:         XTerm
xterm*WaitForMap:       true
xterm*MarginBell:       false
xterm*JumpScroll:       true
```

Die Datei `.Xdefaults`

Wie bereits erläutert, können sich die Angaben einer Resource-Definition auf ein oder mehrere Widgets beziehen. Beginnt die Definition mit einem Stern, so gilt sie für alle Applikationen. Im obigen Beispiel wurde der Zeichensatz für alle OSF/Motif-Programme (nur diese kennen eine Resource namens Fontlist) auf `"fixed"` gesetzt.

Doch nicht nur das Erscheinungbild der einzelnen Applikationen kann der Benutzer beeinflussen, auch die meisten X-Window-Manager lassen sich individuell konfigurieren. Da sehr viele Linux-Anwender wahrscheinlich den `olvwm` benutzen, wird hier nur auf diesen näher eingegangen. Allgemeine Parameter des `olvwm` werden ebenfalls über Resources modifiziert, beispielsweise in der Datei `.Xdefaults`.

```
olvwm.AutoShowRootMenu:      true
olvwm.VirtualSticky:         xclock xeyes Virtual
olvwm.AllowMoveIntoDesktop:  true
```

Konfiguration des `olvwm` innerhalb von `.Xdefaults`

Die Definition der Workspace-Menüs (diese erscheinen beim Klick auf den Hintergund) erfolgt nicht über Ressourcen, sondern über die Datei `.openwin-menu` im Home-Verzeichnis des Benutzers. Hier können beliebig geschachtelte Menüstrukturen definiert werden. Es ist jedoch nicht sinnvoll, diese mit zu vielen Menüpunkten zu überladen. Vielmehr sollten nur die wichtigsten Applikationen, wie ein Terminal-Emulator oder der bevorzugte Editor in ein solches Menü gelegt werden.

```
#
#  openwin-menu
#  OpenWindows root menu file - top level menu
#

"Workspace" TITLE

Programs MENU
    "Programs" TITLE
    "XTerm..."    DEFAULT exec $OPENWINHOME/bin/xterm -sb -sl 255
    "Clock..."    exec $OPENWINHOME/bin/clock
Programs END PIN

Special MENU
    "Special" TITLE
    "Ftp Tool..."  exec /usr/local/bin/ftptool
    "X Archie..."  exec /usr/local/bin/xarchie
    "News ..."     exec /usr/local/bin/xvnews
    "XV..."        exec /usr/local/bin/xv
    "Directory"    DIRMENU
Special END PIN

SEPARATOR

"Exit..."              EXIT
```

Beispiel für `.openwin-menu` Datei

9.10 Speicher-Optimierung

Für Rechner, auf denen nur wenig Hauptspeicher und Plattenplatz zur Verfügung steht, beispielsweise Notebooks, gibt es spezielle Versionen des X-Window Systems für Linux. Dabei wird meistens auf eine Farbdarstellung verzichtet und auf den XFree Mono Server zurückgegriffen. Außerdem wird die Anzahl der X-Clients und Zeichensätze auf das Nötigste beschränkt. Das sogenannte *TinyX*-Paket, benötigt weniger als 4 MB Plattenplatz und erlaubt zusammen mit speziellen Clients wie dem `fvwm` als Window-Manager oder `rxvt` als Ersatz für das normale `xterm` ein Arbeiten unter der grafischen Oberfläche mit 4 MB Hauptspeicher.

Sprachen & Tools

In diesem Kapitel sollen die verschiedenen Compiler, Interpreter, und Tools zur Programmentwicklung unter Linux erläutert und ihre wichtigsten Merkmale genannt werden.
Eine besondere Eigenschaft von Linux ist die große Anzahl von frei verfügbaren Programmiersprachen, die in den verschiedenen Linux-Installationspaketen im Binärformat enthalten sind. Die meisten dieser Werkzeuge stehen kommerziellen Systemen in nichts nach.

10.1 Sprachen

Die wichtigste Sprache eines UNIX-Systems ist unbestritten C. UNIX selbst wurde mit all seinen Utilities von Anfang an in C entwickelt und so ist es nicht verwunderlich, daß ein C-Compiler das zentrale Element einer UNIX-Entwicklungsumgebung darstellt.
Bei kommerziellen UNIX-Systemen war bisher in der Regel ein C-Compiler im Lieferumfang enthalten. Dies hat sich jedoch inzwischen geändert. Gerade die neueren PC-UNIX Implementationen werden mittlerweile in einer Anwenderversion ohne C-Compiler und ohne Netzwerk-Unterstützung angeboten. Die Vollversion, in der diese Komponenten enthalten sind, ist dann natürlich erheblich teurer.
Die von Linux unterstützten Sprachen reichen von C, C++ und Objective-C über Lisp und Prolog bis hin zu Smalltalk und Forth. Auch traditionellere Sprachen wie Fortran oder APL stehen unter Linux zur Verfügung. Selbst BASIC Programme können mit zwei

verschiedenen Interpretern entwickelt werden. Ein Modula-3-Compiler steht kurz vor der Vollendung.
Sobald der GNU-ADA-Compiler stabil läuft, ist eine Linux-Version zu erwarten. Es dürfte in absehbarer Zeit wohl kaum eine Sprache geben, die nicht unter Linux verfügbar wäre.

10.2 C/C++-Compiler

Als C-Compiler wird unter Linux der GNU-C-Compiler `gcc` der Free Software Foundation benutzt. `gcc` erzeugt auf Wunsch optimierten Code und steht auf zahlreichen UNIX-Plattformen zur Verfügung, wodurch die Portierung von Software erheblich vereinfacht wird. Neben C nach dem ANSI- oder K&R-Standard wird vom GNU-Compiler auch C++ und Objective C unterstützt. Besonders erwähnenswert sind die detailierten Fehlermeldungen dieses Compilers bei syntaktischen Fehlern.

10.3 Pascal, Fortran, Simula, Modula-2

Auf dem GNU C-Compiler basieren auch die Pascal- und Fortran- nach-C-Konverter. Diese lesen eine bestehende Pascal- oder Fortran-Quelldatei ein und erzeugen daraus ein C-Programm, das dann mit dem C-Compiler übersetzt wird. Durch geschickte Integration in das Make-System kann dies so erfolgen, daß der Programmierer den Zwischenschritt über C gar nicht bemerkt.
Der Pascal-nach-C-Konverter ist sogar in der Lage, neben Standard-Pascal die Dialekte mehrerer Pascal-Varianten, wie Turbo Pascal bis zur Version 5 oder Macintosh Pascal zu übersetzen.

Simula

Auch der Simula-nach-C Konverter arbeitet nach dem oben beschriebenen Prinzip.

Modula-2

Die *Gesellschaft für Mathematik und Datenverarbeitung* (GMD) hat mittlerweile ihre Modula-2-Entwicklungsumgebung MOCKA auf Linux portiert. Interessant ist, daß der Compiler selbst in Modula-2 implementiert wurde. Entsprechende Quelltexte liegen dem Paket bei. MOCKA erzeugt, im Gegensatz zu den oben

genannten Tools, direkten Object-Code. Auch die Einbindung von C-Routinen über sogenannte *Foreign Moduls* wird unterstützt.

10.4 Lisp/Prolog

Da die Sprachen, die im Bereich der wissensbasierten Systeme und der KI verwendet werden, an den Hochschulen eine wichtige Rolle spielen, existieren hier eine ganze Reihe von verfügbaren Implementationen. Lisp spielt hier eine besondere Rolle, da es nicht zuletzt als Programmiersprache des GNU Emacs relativ weite Verbreitung gefunden hat.
Eine Common Lisp Implementation mit objektorientierter Erweiterung (einem Subset von CLOS) namens *clisp* ist schon im SLS-Paket enthalten.
Auf dem niederländischen FTP-Server `swi.psy.uva.nl` existiert eine umfangreiche Prolog-Implementierung der Universität von Amsterdam, die sich recht einfach unter Linux übersetzen läßt.

10.5 Tcl

Tcl ist, wie im Kapitel über Toolkits (Kap 9.6) bereits erwähnt, eine Interpretersprache, die als C-Library vorliegt. Sie wird beispielsweise in Anwendungen, die eine Scriptsprache benötigen, integriert und dabei mit Hilfe von C-Routinen um zusätzliche Befehle erweitert. Es lassen sich jedoch auch komplette Anwendungen mit Tcl entwickeln. Ein Tcl-Compiler zur Steigerung der Performance befindet sich momentan in der Entwicklung.

Tcl bietet alle Konstrukte, die man von herkömmlichen Programmiersprachen kennt, wie Schleifen, Bedingungen oder Prozeduren. Darüber hinaus wurden einige zusätzliche Features implementiert, die extrem elegante und kurze Programme ermöglichen. Besonders interessant sind beispielsweise die

assoziativen Arrays oder die Möglichkeiten zur Listenverarbeitung.

Beim Entwickeln einer Anwendung mit Tcl bieten sich grundsätzlich zwei Vorgehensweisen an: Zunächst kann eine in C entwickelte Applikation durch den Tcl-Interpreter um eine Scriptsprache erweitert werden. Die Anbindung an die Anwendung erfolgt dann durch Definition zusätzlicher Tcl-Befehle. Diese Vorgehenweise dürfte vor allem bei schon existierenden Programmen sinnvoll sein.
Bei einer Neuentwicklung könnte die Hauptapplikation auch in Tcl implementiert werden. Funktionen, die nicht verfügbar sind lassen sich als Tcl-Erweiterungen in C realisieren und über Tcl ansprechen. In diesem Fall bietet es sich an, auch die Benutzeroberfläche mit dem Tcl-basierten Toolkit Tk zu verwirklichen.
Es existieren schon viele frei verfügbare Tcl-Spracherweiterungen. Diese ermöglichen beispielsweise den Zugriff auf SQL Datenbanken oder UNIX-Systemaufrufe zur Netzwerkprogrammierung.
Ein FTP-Server, auf dem neben zahlreichen Tcl-Erweiterungen auch viele fertige Tcl/Tk Applikationen zu finden sind, ist `harbor.ecn.purdue.edu`.

10.6 awk, gawk

Bei `awk` handelt es sich um ein traditionelles UNIX-Tool, das die Erstellung kleinerer Scripts zur Verarbeitung von Strings und Textdateien erlaubt. Der Name des Programms setzt sich aus den Anfangsbuchstaben der Autoren Aho, Weinberger und Kernighan zusammen. Da sich die Interpretersprache von `awk` syntaktisch sehr stark an C anlehnt, ist sie relativ leicht erlernbar.
`awk` kennt keinen Unterschied zwischen numerischen und String-Variablen und verlangt auch keinerlei Variablendeklaration. Für kleinere Projekte kann `awk` auch als Prototyping-Tool eingesetzt werden. Jedoch sollte die Größe eines `awk`-Programms die Grenze von ca. 200 Zeilen nicht überschreiten.

Folgendes Script summiert die Größe aller im aktuellen Verzeichnis enthaltenen Dateien auf und gibt diese auf dem Bildschirm aus.

```
linux1:/> ls -l | awk '{sum += $5} END {print "Summe : " sum}'
```

Beispiel einer awk-Anwendung

Unter Linux steht jedoch nicht das Original Utility zur Verfügung, sondern die Implementierung der Free Software Foundation namens GNU awk (gawk). Kommerzielle UNIX-Systeme enthalten neben awk meist auch noch eine erweiterte Version mit dem Namen nawk.

10.7 Perl

Auch Perl ist eine Interpretersprache. Sie vereint die wichtigsten Features von sed, awk und den üblichen UNIX-Shells und eignet sich ebenfalls besonders zur Bearbeitung von Textdateien. Ein Vorteil gegenüber den klassischen Werkzeugen ist die Möglichkeit, mehrere Dateien gleichzeitig zu öffnen.
Perl erlaubt die Verwendung von Listen und Assoziativ-Arrays. Die Größe solcher Datenstrukturen ist nur durch den zur Verfügung stehenden Speicherplatz beschränkt. Daher können auch relativ große Datenmengen verarbeitet werden. Natürlich kennt Perl auch Schleifen, Abfragen, Unterroutinen, Rekursion und Regular Expressions. Zur Datenausgabe können verschiedene Formatanweisungen benutzt werden, was die Erstellung von übersichtlichen Reports und Tabellen ermöglicht.
Die Fehlersuche wird duch einen integrierten Debugger erleichtert, mit dessen Hilfe beispielsweise Breakpoints gesetzt werden können oder das Programm Schritt für Schritt durchlaufen werden kann.
Perl eignet sich auch zur Systemadministration, da sich Scripts erstellen lassen, die mit Root-Berechtigung laufen, ohne eine Sicherheitslücke darzustellen. Bemerkenswert ist außerdem, daß beinahe alle Routinen der Standard C-Bibliothek und des UNIX-Kernels unter Perl direkt benutzt werden können. Dies schließt auch Funktionen zur Netzwerkprogrammierung über Sockets ein.

Momentan wird an einer Erweiterung gearbeitet, die auch die Erstellung grafischer Benutzerschnittstellen und die Anwendung objektorientierter Programmiertechniken erlauben soll.

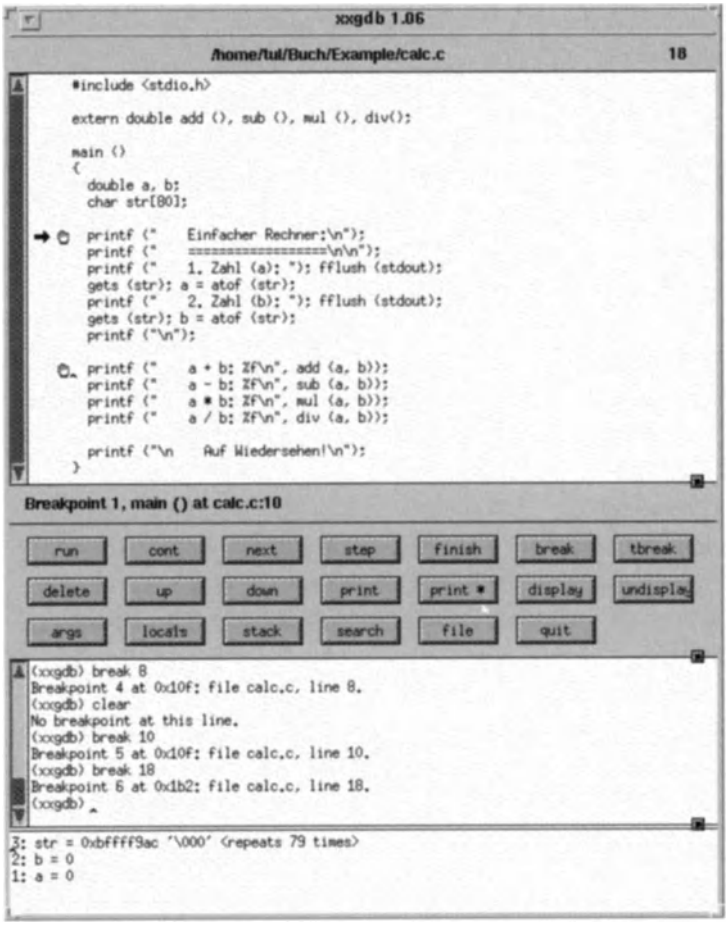
10.8 Editoren

Ein Compiler alleine macht bekanntlich keine vollständige Entwicklungsumgebung aus. Dazu kommt noch ein passender Editor und ein symbolischer Debugger. Ein diese Komponenten integrierender Editor ist GNU Emacs. Er kann neben zahlreichen Utilities wie `grep` oder RCS sowohl den C-Compiler als auch den GNU Debugger aufrufen, so daß der komplette Entwicklungszyklus innerhalb des Editors durchlaufen werden kann. Neben dem GNU Emacs stellt das Linux-System noch zahlreiche weitere Editoren zur Verfügung, die jedoch in einem eigenen Kapitel (Kap. 11) vorgestellt werden.

10.9 GNU-Debugger (GDB)

Beim GNU Debugger handelt es sich um einen leistungsfähigen symbolischen Debugger, mit dem C-, C++- und Modula-2-Programme bearbeitet werden können. Er bietet sämtliche Funktionen, die von einem solchen Werkzeug erwartet werden.
Programme können schrittweise abgearbeitet werden. Breakpoints ermöglichen die Unterbrechung des Ablaufs an definierten Stellen und durch sogenannte Watchpoints wird das Programm bei Veränderung bestimmter Speicherbereiche unterbrochen. Auf Wunsch kann auch der Inhalt ausgewählter Variablen oder Objekte ständig ausgegeben werden.
Neuere Versionen des GDB erlauben darüber hinaus sogenanntes Remote-Debugging. Darunter versteht man die Möglichkeit, daß der Debugger auf einer anderen Maschine läuft als das zu debuggende Programm. Die Verbindung zwischen beiden Rechnern erfolgt entweder über eine serielle Schnittstelle oder ein Netzwerk. Unter Linux kann man sogar den Betriebssystemkern mittels GDB analysieren und laufende Prozesse nachträglich mit dem Debugger bearbeiten.

Der GNU Debugger verfügt zunächst nur über eine einfache kommandozeilenorientierte Benutzerschnittstelle. Es gibt jedoch grafische Frontends, die eine komfortable Mausbedienung des Debuggers ermöglichen. Diese starten den eigentlichen GDB als zweiten Prozeß und leiten dessen Ein- und Ausgabekanal um. Auf diese Weise werden Benutzereingaben simuliert und Debuggerausgaben in verschiedene Fenster umgelenkt. Ein Beispiel für ein solches Frontend ist `xxgdb`.



`xxgdb`, ein grafisches Frontend für `gdb`

Im oberen Fenster wird die aktuelle Stelle im Quelltext angezeigt. Der mittlere Bereich enthält eine Reihe von

Kommandobuttons, über die grundlegende Debuggerfunktionen aufgerufen werden können. Im unteren Fenster kann der Benutzer außerdem Textkommandos eingeben, falls die Eingabe über Tastatur einfacher erscheint.
GDB ermöglicht die permanente Ausgabe von Variableninhalten. Diese werden in einem vierten Fensterbereich angezeigt. Die Auswertung eines Ausdrucks, der im C-Quelltext vorkommt, kann auf einfache Weise mittels Maus erfolgen. Dazu wird der Ausdruck im Quelltextfenster selektiert. Nun kann dieser über einen Mausklick auf den Print-Button ausgewertet und das Ergebnis im zweiten Fenster ausgegeben werden.
Die Definition eines Breakpoints erfolgt in ähnlicher Weise. Man positioniert den Text-Cursor im Quelltextfenster auf die gewünschte Stelle und betätigt den entsprechenden Button. Breakpoints werden durch das Symbol einer flachen Hand angezeigt. Die Markierung der aktuellen Stelle im Quelltext erfolgt durch einen blauen Pfeil.

Eine weitere Möglichkeit, die Benutzung des GNU-Debuggers komfortabler zu gestalten, ist die Integration durch den Emacs-Editor. Ein spezieller Modus dieses Editors erlaubt das Debugging von Programmen innerhalb eines normalen Editorfensters.
Dazu wird dieses in zwei Bereiche aufgeteilt. Im oberen erfolgt die Ausgabe des Quelltextes und die Markierung der aktuellen Stelle duch einen Pfeil. Das untere Fenster dient zur Steuerung des Debuggers und zur Eingabe von Kommandos. Die Benutzung des GDB innerhalb von Emacs hat für den Emacs-Kenner zwar Vorteile, ist aber leider nicht so komfortabel wie die Benutzung spezieller X-basierter Frontends.

gdb im emacs

10.10 Make-Utility

Eine weitere wichtige Komponente der C-Entwicklungsumgebung ist das `make`-Utility. Mit Hilfe dieses Kommandos läßt sich die Übersetzung eines aus mehreren Modulen bestehenden Projektes erheblich vereinfachen.

Dazu werden die Abhängigkeiten zwischen den einzelnen Programmteilen vom Programmierer in einem sogenannten *Makefile* festgehalten. Ändert der Programmierer den Quelltext eines dieser Module, so werden nur die modifizierten Teile und die davon abhängigen Module neu übersetzt.

Die unter Linux benutzte GNU-Variante von `make` enthält einige Optionen, die über das normale `make`-Utility hinausgehen. Sie unterstützt beispielsweise das Revision Control System (RCS),

eine Sammlung von Kommandos, die eine Versionsverwaltung von Quelltexten ermöglichen. Dieses System wird im nächsten Abschnitt näher beschrieben.
GNU-`make` sucht, falls eine Datei nicht im aktuellen Verzeichnis gefunden werden kann, nach einem Unterverzeichnis namens RCS, aus dem die jeweils neueste Version eines Moduls automatisch "ausgecheckt" wird. Dies gilt sogar für das eigentliche Makefile.

10.11 Versions-Kontrolle

Bei der Entwicklung größerer Projekte, die aus einer Vielzahl unterschiedlicher Module bestehen und von denen immer wieder neue Versionen entstehen, ist eine Versionsverwaltung unverzichtbar. Ein derartiges System ist vor allem auch dann sinnvoll, wenn mehrere Programmierer gleichzeitig an einem Projekt arbeiten.
Da sämtliche Versionen eines im Entwicklungszyklus befindlichen Programms gespeichert werden, kann im Bedarfsfall auf die Quelltexte einer längst überholten Programmversion zurückgegriffen werden. Linux bietet neben dem von UNIX System V bekannten Source Code Control System (SCCS) das leistungsfähigere Revision Control System (RCS). Im Normalfall werden die Archivdateien in einem separaten Verzeichnis namens RCS bzw. SCCS abgelegt.
Das RCS stellt zahlreiche Kommandos zur Verfügung, von denen die wichtigsten kurz betrachtet werden sollen.

- **ci** (check in) übernimmt die angegebenen Dateien in das RCS-Verzeichnis und friert somit die vorliegende Version ein. Es wird jedoch nicht bei jedem Checkin die komplette Datei gesichert, sondern nur die durch das Kommando `diff` ermittelte Differenz zur Vorgängerversion. Die Versionsnummer des Moduls wird dabei automatisch erhöht.
- **co** (check out) erzeugt eine nicht modifizierbare Kopie der aktuellen Version oder einer beliebigen Vorgängerversion eines Moduls. Soll ein Modul verändert werden, so muß dies mit der Option `-l` angezeigt werden. Dadurch wird die

Datei vor einer Modifikation durch andere Programmierer geschützt, denn sie kann nur einmal von einem Benutzer ausgecheckt werden.

- **rlog** zeigt verschiedene Informationen an, wie den Zustand eines Archives und die Versionen der darin enthaltenen Module.

Außerdem stehen noch einige weitere Kommandos zur Verfügung, die beispielsweise die Zusammenführung zweier Versionszweige erlauben oder bestimmte Optionen der Versionsverwaltung steuern. Um eine reibungslose Zusammenarbeit zwischen der Versionsverwaltung und der restlichen Entwicklungsumgebung zu ermöglichen, bieten neuere Versionen des Emacs-Editors einen speziellen Modus zur Bedienung des RCS.

10.12 XWPE

Eine interessante Alternative zu Emacs als Entwicklungsumgebung ist XWPE. Anwender, die bereits mit den Borland Produkten für DOS (Turbo-C oder Turbo-Pascal) vertraut sind, werden sich schnell zurecht finden, da XWPE diesen sehr ähnlich ist. Obwohl es sich um eine zeichenorientierte Applikation handelt, ist eine komfortable Mausbedienung möglich. Der Autor entwickelte dazu eine eigene Terminalemulation mit Farbunterstützung für X-Windows. XWPE ist aber auch auf einem normalen Terminal lauffähig. Allerdings leidet dabei das Erscheinungsbild beträchtlich.

Der integrierte Editor erlaubt die Bearbeitung mehrerer Texte in unterschiedlichen Fenstern. Der Compiler (gcc) wird über Tastendruck oder Mausklick gestartet. Eventuelle Fehlermeldungen erscheinen in einem separaten Fenster. Auch der Debugger (gdb) läßt sich auf diese Weise aktivieren. Breakpoints können direkt im Editor definiert werden.

XWPE als integrierte Entwicklungsumgebung

10.13 Beispiel

Um einen besseren Eindruck vom Umgang mit den oben vorgestellten Entwicklungswerkzeugen zu vermitteln, soll im folgenden ein kleines C-Beispielprogramm vorgestellt werden.
Zunächst wird der Editor gestartet.

lemacs

In diesem Beispiel wird nicht die Standardversion, sondern eine Variante des Emacs-Editors verwendet. Es handelt sich um den `lemacs` der Firma Lucid, der ebenfalls kostenlos erhältlich ist.
Diese Version des Emacs Editors verfügt über eine graphische Menuleiste, die gegenüber der des standard-emacs etwas ansprechender wirkt.

Lucid Emacs, eine Variante des GNU emacs 19

Das Programm besteht aus mehreren Modulen. Nach Eingabe zweier Zahlen sollen diese addiert, subtrahiert, multipliziert und dividiert werden. Das Hauptprogramm enthält nur die beiden Eingabeaufforderungen und die Ausgabe der Ergebnisse. Die Berechnung erfolgt in separaten Modulen. Natürlich hätte dieses Beispiel auch in einer einzigen Datei zusammengefaßt werden können.

Nach der Eingabe des folgenden Quelltextes wird dieser über die Tastenkombination **<Strg-X> <Strg-S>** unter dem Dateinamen "`main.c`" abgespeichert. Diese Funktion kann auch über das File-Menü aufgerufen werden.

```
#include <stdio.h>

extern double add (), sub (), mul (), div();

main ()
{
  double a, b;
  char str[80];

  printf ("    Einfacher Rechner:\n");
  printf ("    ==================\n\n");
  printf ("    1. Zahl (a): "); fflush (stdout);
  gets (str); a = atof (str);
  printf ("    2. Zahl (b): "); fflush (stdout);
  gets (str); b = atof (str);
  printf ("\n");

  printf ("    a + b: %f\n", add (a, b));
  printf ("    a - b: %f\n", sub (a, b));
  printf ("    a * b: %f\n", mul (a, b));
  printf ("    a / b: %f\n", div (a, b));

  printf ("\n    Auf Wiedersehen!\n");
}
```

Die Datei `calc.c`

Nun können auch die anderen Module eingegeben werden. Eine neue Datei wird über die Tastenkombination **<Strg-X> <Strg-F>** geöffnet.

```
double add (double a, double b)
{
  return a + b;
}

double sub (double a, double b)
{
  return a - b;
}
```

Die Datei `addsub.c`

```
double mul (double a, double b)
{
  return a * a;
}

double div (double a, double b)
{
  return a / b
}
```

Die Datei `muldiv.c`

Das Projekt besteht also aus drei Dateien. Dies hat auch auf die Gestalt des Makefiles Auswirkungen.

```
CFLAGS = -g
OBJS = calc.o addsub.o muldiv.o

calc: $(OBJS)
        $(CC) -o $@ $(OBJS)
```

Das `Makefile`

Zunächst wird die Standardvariable `CFLAGS` mit der Option `-g` belegt, so daß der Compiler Debug-Code erzeugt. Die neu definierte Variable `OBJS` enthält nach ihrer Deklaration die Namen sämtlicher Objekt-Dateien. Diese Vereinbarung dient nur der Erhöhung der Lesbarkeit und ist nicht unbedingt nowendig. Als Zieldatei (Target) ist das fertig übersetzte und gelinkte Programm `calc` definiert. Um dieses erzeugen zu können, werden die drei Objekt-Dateien benötigt. Das Kommando

```
        $(CC) -o $< $(OBJS)
```

linkt diese nach der Übersetzung zusammen. Wie man erkennen kann, muß keine separate Regel für die Übersetzung einer `*.c` in eine `*.o`-Datei angegeben werden. Diese ist dem Make-Mechanismus implizit bekannt. Bei der Eingabe des Makefiles ist zu beachten, daß vor der Link-Anweisung ein **<TAB>**-Zeichen stehen muß.

Wurde das Makefile gespeichert, so kann das komplette Projekt aus dem Editor heraus übersetzt und gelinkt werden. Dazu dient das Kommando **<Meta-X> compile**. Das Fenster des Editors teilt sich nun. Im unteren Bereich werden die ausgeführten Befehle mitprotokolliert und eventuelle Fehlermeldungen ausgegeben. Im vorliegenden Fall wurde in der Funktion `div` des Moduls `muldiv.c` absichtlich ein Semikolon weggelassen. Bei der Übersetzung dieser Quelldatei wird der Compiler auf einen Fehler stoßen. Mit dem Kommando **<Meta-X> next-error** wird die Fehlerausgabe vom Editor bearbeitet und der Cursor auf die Zeile im Quelltext positioniert, in welcher der Fehler aufgetreten ist. Die erneute Eingabe springt dann zum nächsten Fehler. Nachdem der Fehler beseitigt wurde, kann dann mit **<Meta-X> compile** der Compiler erneut gestartet werden. Natürlich sollte sich der Programmierer die oben erwähnten Kommandos auf eine freie Tastenkombination seiner Wahl legen,

um diese nicht immer komplett eingeben zu müssen. Derartige Definitionen werden am besten in der Datei `.emacs` im Home-Verzeichnis durchgeführt.
Haben sich logische Fehler in das Programm eingeschlichen, so kann der GNU-Debugger eine nützliche Hilfe für die Fehlersuche darstellen. Da das Programm schon mit der Compiler-Option `-g` übersetzt wurde, kann der Debugger sofort gestartet werden.

```
linux2:home/tul> xxgdb calc
```

Zunächst sollten die benötigten Breakpoints gesetzt werden. Dies kann einerseits über die Maus oder andererseits durch die Eingabe eines entsprechenden Kommandos bewirkt werden. Dazu wird der Cursor im Textfenster zunächst auf die entspechende Zeile gebracht, in der ein Breakpoint gesetzt werden soll. Dieser wird dann über einen Klick auf den Break-Button des Debuggers eingerichtet. Oft ist es jedoch einfacher einen Breakpoint über die Kommandozeile zu setzen, vor allem dann, wenn dieser gleich am Anfang einer Funktion stehen soll.

```
gdb> break main
```

stoppt das Programm gleich beim Eintritt in die Funktion main. Sind alle gewünschten Breakpoints spezifiziert, so kann das Programm mittels `run`-Kommando gestartet werden. Wurde der Ablauf an einem Breakpoint unterbrochen, dann ist es möglich, daß die Ausführung in Einzelschritten durchgeführt oder aber der Inhalt einer Variable ausgegeben wird.

```
gdb> print a
```

gibt beispielsweise den Inhalt der Variablen `a` aus, falls sich diese im aktuellen Sichtbarkeitsbereich befindet. Die dauerhafte Ausgabe von Variableninhalten erfolgt über das Kommando `display`.

```
gdb> display b
```

gibt nach jedem GDB-Kommando den Inhalt der Variablen `b` aus.
Will man das nun fehlerbereinigte Programm starten, so kann dies auch aus Emacs erfolgen. Dazu muß jedoch zunächst eine Shell durch das Kommando **<Meta-X> shell** geöffnet werden. Nun können sämtliche UNIX-Kommandos innerhalb des Editors benutzt werden.
Der Vorteil einer im Emacs ausgeführten Shell ist die Möglichkeit, die bereits bearbeiteten Kommandos einfach editieren zu können und sämtliche Editor-Kommandos, wie beispielsweise das Kopieren eines Blocks, auch auf die Shell-Ausgaben anwenden zu können.

10.14 Portieren von Software

Nur ein kleiner Prozentsatz der Programme, die es auf den vielen FTP-Servern für UNIX-Systeme gibt, ist speziell für Linux angepasst (portiert) worden. Da Linux sich jedoch an den wichtigsten Standards orientiert, ist das Compilieren von Programmen, die für andere UNIX-Systeme geschrieben wurden, unter Linux meist kein Problem.

Entpacken

Die Programme sind auf den FTP-Servern meist als komprimierte `tar`-Dateien abgelegt. Das sind Dateien, die mit dem `tar`-Programm erstellt wurden und mehrere Dateien und Verzeichnisse enthalten können. Dies erkennt man normalerweise auch am Namen der Datei. Endet sie auf `.tar.Z`, so bedeutet dies, daß es sich um ein `tar`-Archiv handelt, das mit dem UNIX-Befehl `compress` komprimiert wurde. Die Endung `.tgz` oder `.taz` signalisiert, daß es sich um ein `tar`-Archiv handelt, das mit dem `gzip`-Programm komprimiert wurde.
Um eine solche Datei zu entpacken genügt unter Linux die Eingabe von:

```
linux1:/home/tul> tar xvfz <Datei.tar.Z>
```

Entpacken eines tar-Archivs

Das `tar`-Programm ruft dabei selbstständig das `gzip`-Programm auf. `Gzip` entpackt Dateien, die mit `gzip` selbst oder `compress` komprimiert wurden.
Quelltexte, die als Mail verschickt wurden, befinden sich oft in sogenannten Shell-Archiven, deren Namen normalerweise auf `.shar` enden. Um ein solches Archiv zu entpacken, genügt eine normale Bourne-Shell:

```
linux1:/home/tul> sh <Datei.shar>
```

Entpacken eines shar-Archivs

Dokumentation

Nach dem Entpacken sollten in jedem Fall zunächst eventuell beiliegende README- oder INSTALL-Dateien gelesen werden, die wichtige Hinweise für die Compilation enthalten. Meistens beziehen sich diese auf die unterstützten Plattformen und eventuell auftretende Probleme.
Oftmals liegt dem Paket auch eine ausführliche Benutzerdokumentation als Postscript- oder TeX-Datei bei. Zumindest eine UNIX Manual page dürfte sich in allen Fällen finden lassen.

Makefile/Imakefile

Beim Quellcode eines Programms findet sich eigentlich immer ein Makefile, in dem man wichtige Optionen einstellen kann. Dazu gehören zum Beispiel Compilereinstellungen, die Pfade der Libraries oder das Installationsverzeichnis.
Die Compilation und Installation erfolgt ebenfalls mit Hilfe des Makefiles. Durch den Aufruf von `make` und `make install` wird der C-Compiler bzw. der Linker gestartet und das Programm anschließend installiert. Alle diese Befehle und Optionen werden jedoch meist in den README-Dateien, die in fast jedem Programmpaket enthalten sind, näher erläutert.

Noch einfacher ist die Konfiguration bei X-Windows-Software. X-Windows-Programme enthalten im allgemeinen kein Makefile sondern ein *Imakefile*. Dies ist eine Art systemunabhängiges Makefile. Aus dem Imakefile wird mit dem Befehl `xmkmf` automatisch ein Makefile erzeugt, das schon alle wichtigen Pfade und Optionen des jeweiligen Systems enthält. Bei solchen Programmen reicht es meist aus, `xmkmf`, `make` und `make install` einzugeben.

Configure

Neuere Programmpakete der FSF enthalten oft ein Script mit dem Namen `configure`. Dieses Script wird als erstes aufgerufen, erkennt selbst das Betriebssystem und durchsucht dann das System nach dem C-Compiler, anderen Tools und Libraries.
Die gefundene Information wird gespeichert und beim Compilieren verwendet, so daß keine weiteren Anpassungen nötig sind.

Manuelle Nacharbeit

Bricht der Compiler im Make-Vorgang mit einer Fehlermeldung ab, so muß man einige Anpassungen selbst durchführen. Bei Programmen, die kein `configure` Script benutzen kommt dies häufig vor. Mit etwas Übung sind diese Anpassungen jedoch meist in kurzer Zeit durchgeführt. Eine typische Fehlermeldung lautet zum Beispiel:

```
linux2:/home/tul/tmp> gcc bsp.c
bsp.c: In function `main':
bsp.c:3: `errno' undeclared (first use this function)
bsp.c:3: (Each undeclared identifier is reported only once
bsp.c:3: for each function it appears in.)
linux2:/home/tul/tmp>
```

Mögliche Fehlermeldung

Der Compiler kennt also die Definition eines Symbols nicht. Dies kann mehrere Ursachen haben. Wenn es sich bei dem Symbol um eine Konstante oder ein Makro handelt, so ist diese entweder

unter Linux nicht vorhanden oder das Headerfile, in dem die Definition steht, wurde nicht mit `#include` eingebunden.
Das gleiche gilt für C-Funktionen, deren Prototypen in Headerfiles stehen. Zunächst sollte man versuchen, sich mittels Manual page einen Überblick über die Funktion und deren Parameter zu verschaffen.
Um festzustellen, ob eine Funktion oder ein anderes Symbol unter Linux existiert, kann man auch die Systemincludes, also die C-Headerfiles der Linux C-Library durchsuchen.
Diese Headerfiles befinden sich im Verzeichnis `/usr/include` und können zum Beispiel mit dem folgenden Kommando durchsucht werden:

```
find /usr/include -name "*.h" -exec grep "symbol" {} \; -print
```

Suche eines unbekannten Symbols in den Quelltexten

Auf diese Weise werden mit dem `find`-Befehl alle Headerfiles im Verzeichnis `/usr/include` und allen Unterverzeichnissen gesucht und diese dann mit dem Befehl `grep` nach dem Symbol durchforstet.
Wenn es sich bei dem undefinierten Symbol um eine Funktion handelt, die auf vielen UNIX-Systemen verschieden ist, so sind häufig schon im Quellcode Alternativen enthalten, die über eine Compileroption wie -DSYSV oder -DBSD aktiviert werden können. Im Quellcode sieht dies dann beispielsweise folgendermaßen aus :

```
#ifdef SYSV
       funktion_a (x, y, z);
#else
       funktion_b (x, y, z);
#endif
```

Bedingte Compilation

Muß man dennoch selbst Änderungen im Quellcode durchführen, so sollte man die Änderungen in `#ifdef linux` Anwesungen einbetten. `linux` ist dabei ein Symbol, das automatisch vom Compiler definiert wird, wenn er unter Linux aufgerufen wird.

```
#ifdef linux
        eigene Änderungen
#else
        alter code
#endif
```

Bedingte Compilation für Linux-Varinate

Falls der Compiler keine Fehlermeldungen anzeigt, jedoch der Linker Fehlermeldungen über undefinierte Symbole ausgibt, so ist das Programm nm eine wertvolle Hilfe. Es gibt eine Liste der definierten und verwendeten Symbole von Objektdateien und Libraries aus.
Zusammen mit grep, zum Durchsuchen der Liste der undefinierten Symbole, kann so festgestellt werden, wo das Symbol verwendet wird und in welcher Library es enthalten ist.

```
linux1:/home/tul>nm prog1.o | grep "symbol"
```

Suche nach einem Symbol in einer Objektdatei

Mit etwas Übung wird man kleinere Programme in wenigen Minuten so anpassen können, daß sie sich unter Linux compilieren lassen.

Archivierung

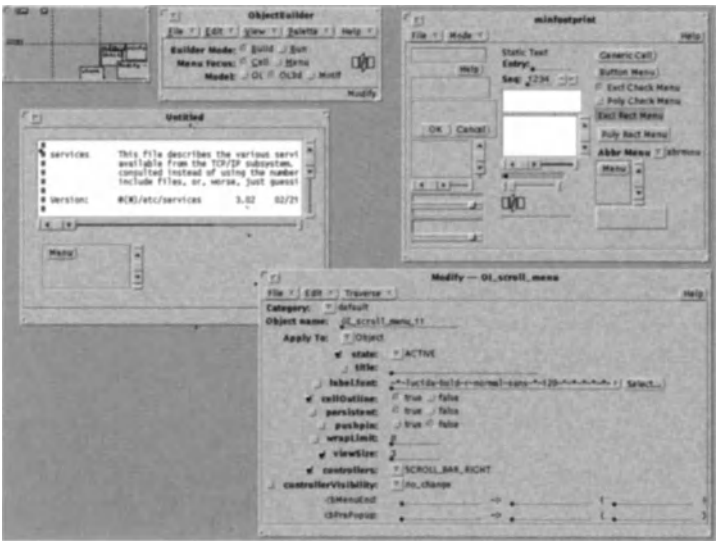
Hat man eine Reihe von Änderungen an dem Quellcode eines Programms gemacht, so sollte man diese Änderungen mit dem diff-Befehl in eine Datei sichern. Dann kann man jederzeit mit Hilfe des patch-Programms die Änderungen auf das ungeänderte Programm anwenden und diese Änderungen auch anderen Benutzern zugänglich machen.

diff, patch

10.15 Interface - Builder

Die Erstellung grafischer Benutzeroberflächen kann ohne entsprechende Werkzeuge recht mühsam sein. Die Firma *Parc Place* aus Sunnyvale vertreibt ein sehr interessantes Pro-grammpaket zur interaktiven Gestaltung derartiger Benutzerschnitt-

stellen. Der *Object Builder* erlaubt die direkte Manipulation der grafischen Objekte mittels Maus und erzeugt C++ Quelltext. Zusammen mit der *Object Library* können Oberflächen nach OpenLook oder Motif-Konvention generiert werden. Die Auswahl des bevorzugten Look and Feels erfolgt über eine Kommandozeilenoption. Die beiden Produkte sind für alle gängigen UNIX-Plattformen verfügbar. Die Anschaffungskosten belaufen sich im allgemeinen auf mehrere tausend D-Mark. Nur die Linux-Version ist kostenlos erhältlich und darf frei kopiert werden. Für den ernsthaften Einsatz sollten jedoch die zugehörigen Handbücher greifbar sein. Diese sind direkt bei Parc Place erhältlich.



Der kostenlose Interface Builder von Parc Place

Anwendungen

Da für Linux mittlerweile eine unüberschaubare Fülle an Anwendungsprogrammen existiert, kann an dieser Stelle nur eine kleine Auswahl vorgestellt werden.

11.1 Arbeitsumgebung G.R.E.A.T.

Eine noch recht neue Entwicklung, die im Rahmen von Linux/CV implementiert wird ist G.R.E.A.T. Diese Abkürzung steht für *Graphical Environment and Desktop* und stellt dem Anwender eine komfortable Umgebung zur Verwaltung aller Systemkomponenten zur Verfügung. Wichtige Applikationen können in einer Iconleiste am Bildschirmrand plaziert werden. Animierte Icons und Sound-Ausgaben sorgen für ein ansprechendes Feedback. Mittels Drag & Drop können Dateien aus dem Dateimanager auf die Icons-Leiste geschoben werden, was zum automatischen Start der entsprechenden Applikation führt. G.R.E.A.T. benutzt das OSF/Motif-Widgetset und soll als Integrationsplattform für zukünftige Linux-Anwendungen dienen. Als weitere Komponenten sind unter anderem ein Multimedia Mailprogramm und Tools zur komfortablen Systemverwaltung geplant.

G.R.E.A.T. Umgebung

11.2 Editoren

Eine der wichtigsten Komponenten eines Computersystems dürften Programme zur Bearbeitung von Texten sein. Unter Linux stehen beinahe ein Dutzend solcher Editoren zur Verfügung. Einige davon sind auf jedem UNIX-System vorhanden, andere müssen auf kommerziellen Systemen nachträglich erworben werden.

vi

Der Standardeditor eines jeden UNIX-Systems ist wohl der `vi`. Da es sich dabei um ein recht betagtes Programm handelt, ist es nicht weiter verwunderlich, daß der Benutzerkomfort an einigen Stellen zu wünschen übrig läßt. Die meisten Benutzer dürfte stören, daß `vi` zwischen einem Kommando- und Eingabemodus unterscheidet. Dieser Ansatz bietet zweifelsohne auch einige Vorteile, an die man sich jedoch zunächst gewöhnen muß. So ist es beispielsweise möglich, die letzte Kommandokette zu wieder-

holen oder die nächsten drei Worte in einem Schritt durch ein neues zu ersetzen.
Da die Quelltexte des vi-Editors nicht frei erhältlich sind, kommt unter Linux ein vi-Clone zum Einsatz. Doch auch bei den Clones gibt es inzwischen schon wieder zwei Alternativen, nämlich elvis und vim. Beide bilden (fast) alle Kommandos des Originals nach und bieten einige darüberhinausgehende Erweiterungen.

sed

sed ist eigentlich kein Editor im gewöhnlichen Sinn, er wird vielmehr als Stream-Editor bezeichnet. Darunter versteht man die Möglichkeit, einen Datenstrom mittels Kommandos aus einer Steuerdatei modifizieren zu können. Die Befehle des sed sind weitgehend kompatibel zu denen des vi, werden jedoch nicht interaktiv vom Benutzer eingegeben, sondern aus einer Datei oder der Kommandozeile gelesen. sed wird oft in UNIX-Shell-Scripts oder zur Bearbeitung sehr großer Dateien benutzt, die von normalen Editoren nicht geladen werden können.

joe

joe ist die Abkürzung für "Joe's own Editor". Dieser Editor dürfte vor allem bei DOS-Umsteigern beliebt sein, da er sich bezüglich der Tastenbelegung stark an die bekannte PC-Textverarbeitung Wordstar bzw. den Turbo-Pascal-Editor anlehnt. Es gibt außerdem auch keinen getrennten Kommando- und Editiermodus. Markierte Textblöcke werden invers dargestellt, was unter UNIX nicht immer selbstverständlich ist. Joe kann den Bildschirm in mehrere Fenster aufteilen, Absätze formatieren und kennt einen Wortumbruch-Modus.

xedit

xedit wird mit dem X-Windows-System ausgeliefert und läuft daher, im Gegensatz zu den bisherigen Editoren, auch nicht in

einer ASCII-Umgebung. Obwohl `xedit` unter einer grafischen Benutzeroberfläche läuft, bietet er nicht den Komfort den man erwarten könnte. Die zur Verfügung stehenden Kommandos beschränken sich im wesentlichen auf die durch das Athena-Text-Widget vorgegebenen Möglichkeiten. Daher wird auch dieser Editor nicht besonders viele Anwender haben. Für einfache Aufgaben dürfte er dennoch ausreichen.

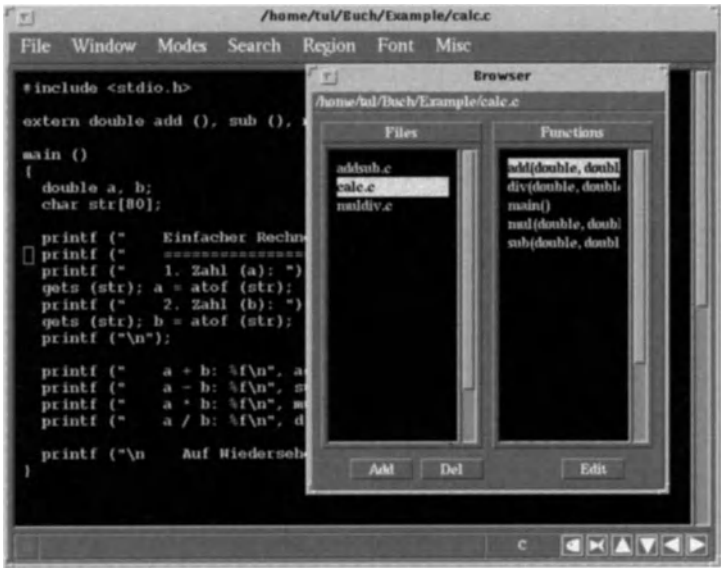
axe

Auch `axe` (*engl.* Axt) ist ein Editor, der nur unter X-Windows läuft. Er benutzt ebenfalls das Athena-Text-Widget, bietet jedoch weit mehr Funktionalität als `xedit`. Der Benutzer kann beliebig viele Texte in verschiedenen Fenstern öffnen, Dateien interaktiv über einen Dateiauswahldialog öffnen und speichern, Textbereiche suchen oder ersetzen oder Absätze formatieren. Eine einfache Online-Hilfe gibt im Bedarfsfall schnell Auskunft über die vorhandenen Kommandos. Mit `axe` steht unter Linux ein vielseitiger und benutzerfreundlicher Editor für X-Windows zur Verfügung.

xcoral

Leider noch nicht ganz fertiggestellt ist der Editor `xcoral`. Eine Vorversion ist aber bereits verfügbar und durchaus einsetzbar. Auch `xcoral` benötigt die X-Windows-Umgebung. Obwohl kein Standard-Toolkit Verwendung findet, ist das Erscheinungsbild sehr ansprechend. Er verfügt neben einer Menüleiste auch über einen Scroll-Bar. `xcoral` ist vor allem für C- und C++-Programmierer interessant, da er auch einen Funktions- bzw. Klassenbrowser integriert hat. Beim Start werden sämtliche C- bzw. C++-Dateien gescannt und alle darin enthaltenen Bezeichner in einer alphabetisch sortierten Liste angezeigt. Von diesem Browser aus kann dann an die entsprechende Stelle im Quelltext weiterverzweigt werden. Auf diese Weise wird vor allem die Einarbeitung in schon vorhandene Quelltexte erleichtert. Beim Schreiben neuer Programme steht optional ein

entsprechender Modus zur Verfügung, der für eine einheitliche Formatierung sorgt und auf Wunsch Funktions- und Klassenschablonen erzeugt. Bezüglich der Tastaturkommandos lehnt sich `xcoral` sehr stark an den bekannten Emacs-Editor an.



xcoral

asedit

`asedit` benutzt das Motif-Widget-Set was sich vor allem in einer konsistenten Oberfläche bemerkbar macht. Dieser Editor bietet zwar die wichtigsten Basiskommandos, wie Texte Laden und Speichern oder Suchen und Ersetzen, ist sonst jedoch eher spartanisch. Besonders erwähnenswert ist noch die integrierte Hypertext-Hilfe.

Emacs und Varianten

Der populärste Editor für UNIX dürfte wohl Emacs in all seinen Varianten sein. Dies liegt wohl nicht zuletzt daran, daß er sowohl auf reinen ASCII-Terminals als auch in einer grafischen X-Windows-Umgebung läuft. `Emacs` wurde vom Gründer der FSF

Richard Stallmann entwickelt und steht auf nahezu allen UNIX-Platformen zur Verfügung. Außerdem sind spezielle Versionen für VMS und MS-DOS verfügbar.

Emacs ist ein nicht modaler, komfortabler Fullscreen-Editor. Der Benutzer kann verschiedene Texte gleichzeitig im Speicher halten und sich diese in getrennten oder geteilten Fenstern anzeigen lassen. In regelmäßgen Abständen wird eine automatische Sicherheitskopie aller Puffer erstellt und eine automatische Reorganisation des Speichers durchgeführt.

Emacs ist großteils in LISP implementiert, was ihn außerordentlich konfigurierbar macht. Der Benutzer kann, entsprechende LISP-Kenntnisse vorausgesetzt, nicht vorhandene Features selbst implementieren. Von dieser Möglichkeit haben auch schon zahlreiche Benutzer Gebrauch gemacht, so daß mittlerweile eine fast unüberschaubare Menge an Emacs-Erweiterungen existiert.

Doch das Emacs-Paket ist aufgrund seiner Mächtigkeit relativ umfangreich. Die Basisinstallation benötigt etwa 15 MB Speicher auf der Festplatte. Darin enthalten ist dann aber auch eine ausführliche Hyptertext-Hilfe, die nicht nur eine Beschreibung des Editors selbst, sondern auch der meisten GNU-Pakete enthält. Besonders bemerkenswert ist auch die unbeschränkte Undo-Funktion. So kann jede Änderung des Puffers per Tastendruck rückgängig gemacht werden.

Emacs stellt zahlreiche sogenannte Major-Modes zur Verfügung, die den Editor optimal an die jeweilige Aufgabe anpassen. Derartige Modi unterstützen den Programmierer oder Autor bei der Formatierung seines (Quell-)Textes. So steht beispielsweise ein Text-Modus zur Verfügung, der einen automatischen Zeilenumbruch erlaubt.

Spezielle Modi für Programmiersprachen

Im C- oder Lisp-Modus werden geschachtelte Anweisungen entsprechend ihrer Tiefe automatisch eingerückt. Bei der Eingabe einer schließenden Klammer springt der Cursor kurz zum entsprechenden öffnenden Klammersymbol und macht auf diese Weise die Hierarchie deutlich. Schlüsselwörter oder Kommentare werden auf Wunsch in einer anderen Schrift angezeigt. Es dürfte wohl kaum eine Programmiersprache existieren, die nicht von Emacs unterstützt wird.

Ein sogenannter `dired`-Modus erlaubt die Verwaltung von Dateien und Verzeichnissen des Filesystems. Man kann Dateien löschen, kopieren oder umbenennen und in den Editor laden. Ein Hexadezimal-Modus ermöglicht die Bearbeitung von Binärdateien. Ein in Emacs-Lisp geschriebenes Mathematik-Paket löst lineare Gleichungssysteme, vereinfacht Terme und ermöglicht symbolische Differentiation.

Darüber hinaus kann Emacs auch als komfortables Frontend für bereits existierende Anwendungen dienen. Ein spezieller Modus (`rmail`) erlaubt das Empfangen und Senden von Mails, ein anderer (`gnus`) die Verwaltung von Nachrichten aus dem USENET.
Für Programmierer interessant ist die Schnittstelle zum GNU-C-Compiler und Debugger. Emacs dient somit auch als Entwicklungs-Shell. Auf Tastendruck kann der aktuelle Quelltext übersetzt werden. Die Ausgabe des Compilers wird dabei in ein separates Fenster umgelenkt. Außerdem kann man im Quelltext von Fehler zu Fehler springen und im Bedarfsfall den Debugger starten.

Emacs ist extrem konfigurierbar. Jede Taste kann an eine beliebige LISP-Funktion gebunden werden. Läuft er in einer grafischen Umgebung, so steht, ab Version 19, neben einer erweiterbaren Menüleiste ein Scroll-Bar zur Verfügung, der die aktuelle Position im Text anzeigt. Außerdem können einzelne Textstellen durch eine spezielle Farbe oder einen anderen Zeichensatz hervorgehoben werden.

Emacs in der Version 19

Doch Emacs hat auch etwas für die Unterhaltung zu bieten. Neben der animierten Lösung des Problems *Türme von Hanoi* steht Weizenbaums elektronischer "Psychiater" *ELIZA* Frage und Antwort.

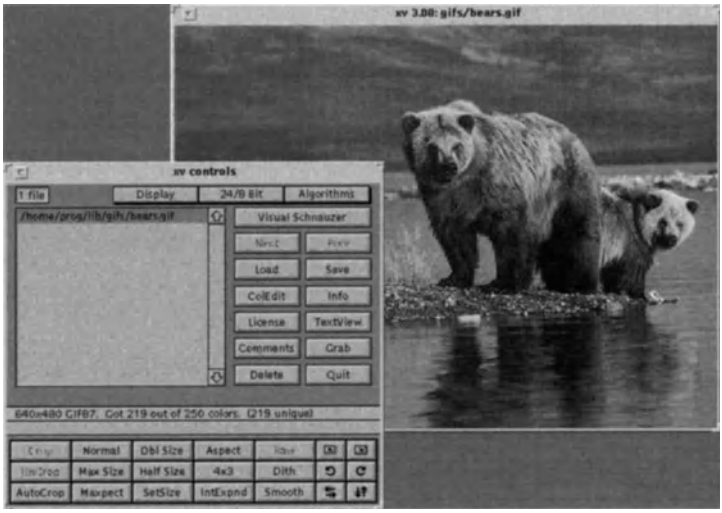
Eine populäre Emacs-Variante, die von der Firma Lucid entwickelt wurde und auf einer frühen Version des Emacs 19 basiert, ist der sogenannte `lemacs`. Er unterscheidet sich hauptsächlich im optischen Erscheinungsbild und der Syntax spezieller Lisp-Funktionen.

Mit dem Erscheinen von Emacs in der Version 19 verloren andere Emacs-Versionen wie `xemacs` oder `epoch` an Bedeutung. Daher soll auf diese hier nicht näher eingegangen werden.

11.3 Grafikprogramme

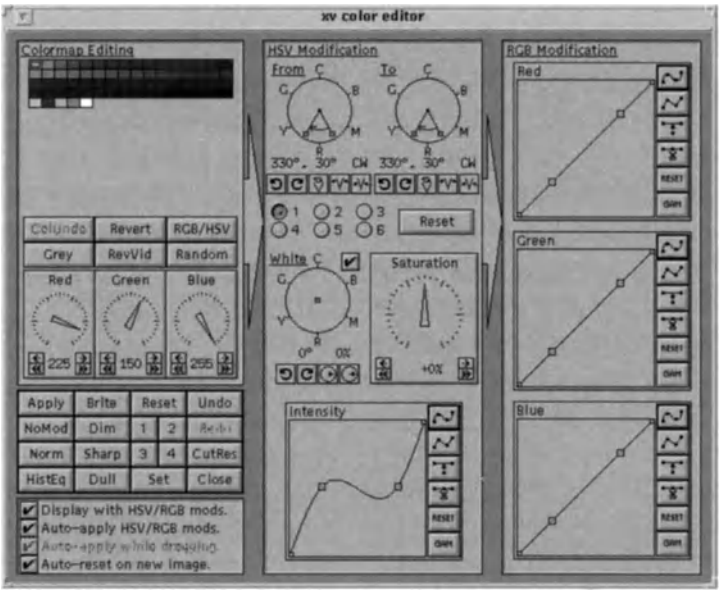
xv

Ein sehr leistungsfähiges und in der UNIX-Welt weitverbreitetes Programm zur Darstellung und Manipulation von Bildern aller Art ist xv von J. Bradley. Es verfügt über eine sehr ausgereifte Oberfläche und ist trotz seiner Mächtigkeit leicht bedienbar. xv lädt und speichert alle gängigen Grafik-Formate wie GIF, TIFF, Postscript, JPEG, PBM und andere.
Die Auswahl des zu ladenden Bildes erfolgt über ein Dateiauswahlmenü oder ab Version 3.00 über einen integrierten Datei-Manager, der die identifizierten Dateien auch als Mini-Bilder darstellen kann.



xv mit File-Browser

Ein geladenes Bild kann in der Größe frei verändert werden. Ein Modul zur Bildbearbeitung erlaubt die Veränderung zahlreicher Parameter wie Helligkeit, Kontrast oder Farbverlauf. Dazu stehen zahlreiche Buttons und Regler sowie mit der Maus modifizierbare Kurvenzüge zur Verfügung.

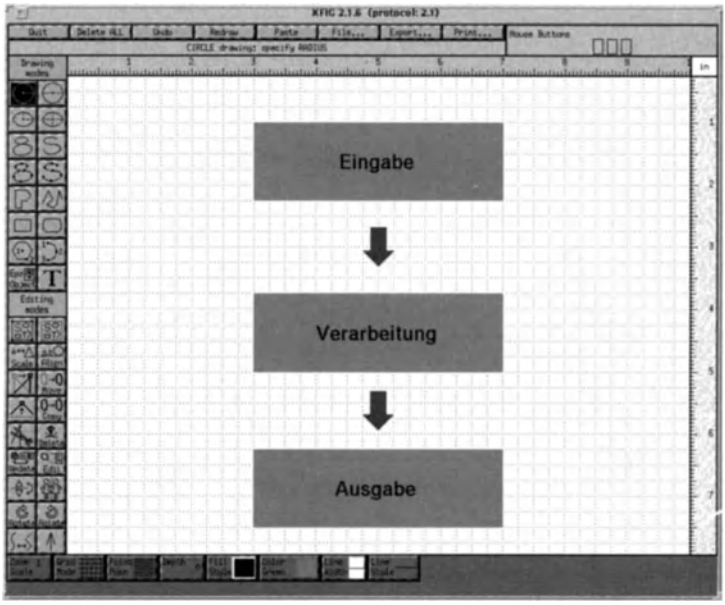


xv Color-Editor

Ein in den `xv` geladenes Bild kann auch auf unterschiedliche Weise verfremdet werden. Dazu können Algorithmen wie "Oil Painting" oder "Amboss" benutzt werden. Außerdem kann dieses in verschiedenen Formen als Hintergrundbild angezeigt werden, das auch nach Verlassen des Programmes erhalten bleibt.

xfig

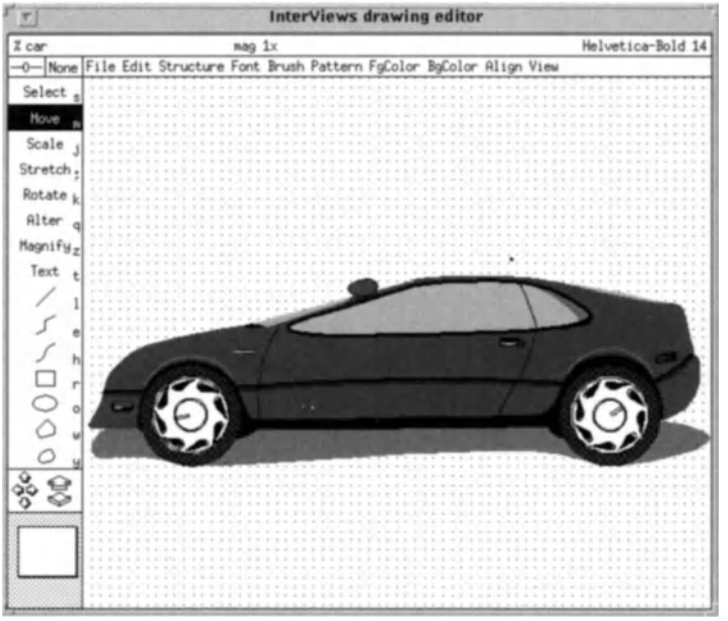
Mit `xfig` steht unter Linux auch ein Programm zur Gestaltung von Vektorgrafiken zur Verfügung. Es werden alle üblichen Zeichenwerkzeuge angeboten. Auch die Einbindung von Texten in unterschiedlichen Zeichensätzen ist möglich. Interessant ist vor allem die Möglichkeit, die erstellten Grafiken in unterschiedlichen Formaten zu exportieren. Dabei werden neben dem eigenen fig-Format auch Standard-Formate wie Postscript, HPGL und TeX in verschiedenen Varianten unterstützt.



Zeichenprogramm `xfig`

idraw

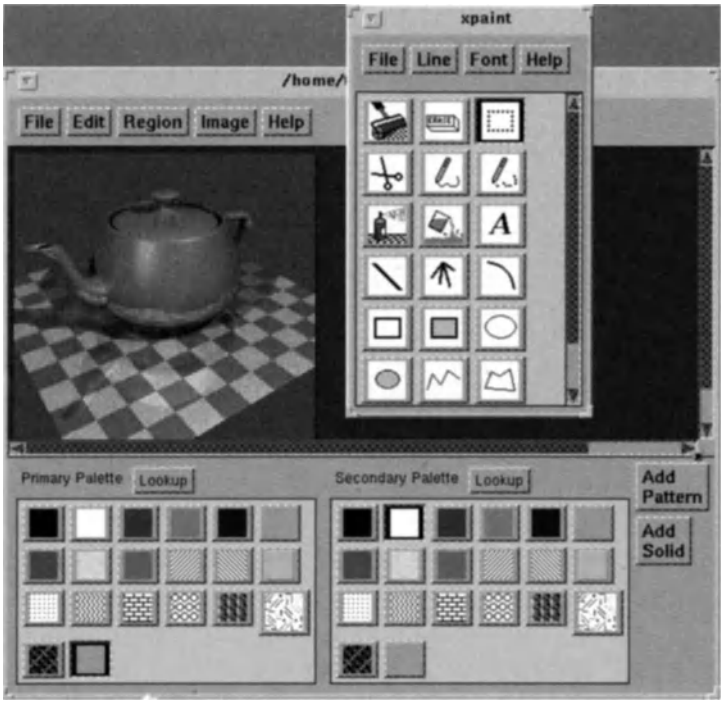
Das InterViews-Paket der Stanford University enthält, wie bereits erwähnt, ein recht leistungsfähiges vektororientiertes Zeichenprogramm namens `idraw`. Neben den üblichen grafischen Elementen, wie Linien, Rechtecken, Kreisen und Bezier-Kurven, können auch beliebig rotierte Texte mit unterschiedlichen Zeichensätzen verwendet werden.
Das einzige Dateiformat, das von diesem Programm unterstützt wird ist Postscript. Leider werden aber relativ hohe Anforderungen an die Hardware gestellt, was das Programm für 386er Computer leider uninteressant macht.



Zeichenprogramm idraw

xpaint

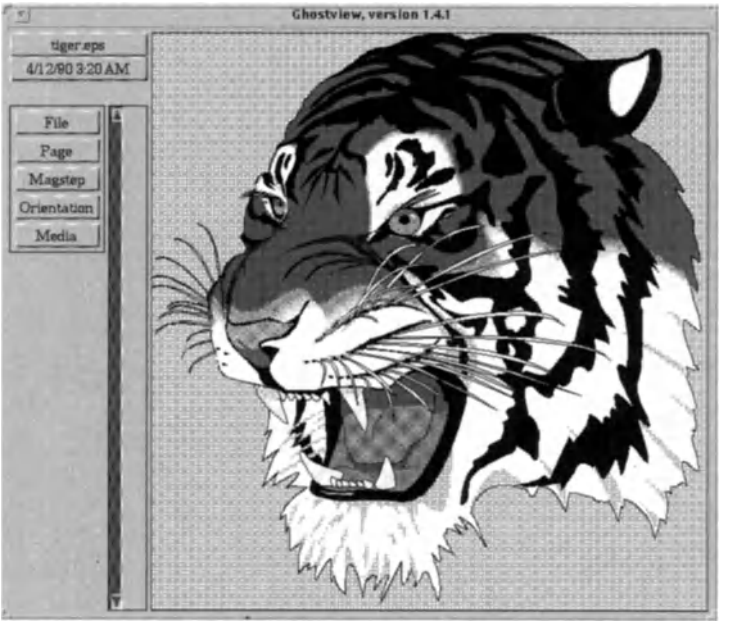
`xpaint` ist ein Programm zur Erstellung und Bearbeitung von Pixel-Grafiken. Es ist ein wenig dem vom Apple-Macintosh bekannten MacDraw nachempfunden. In unterschiedlichen Fenstern können mehrere Grafiken gleichzeitig bearbeitet werden. Verschiedene Werkzeuge ermöglichen das Zeichnen beliebiger Figuren. Dabei werden sowohl einfache Elemente wie Linien und Kreise als auch Freihand-Kurven unterstützt.
Ein Füllmuster-Editor erlaubt die Erstellung zusätzlicher Muster. Sehr praktisch ist die in ihrer Vergrößerung einstellbare Lupenfunktion, die einen Bildausschnitt in einem separatem Fenster vergrößert darstellt. Bemerkenswert ist, daß sämtliche Zeichenoperationen auch in diesem Fenster anwendbar sind. Eine übersichtliche Auswahl des aktuellen Zeichensatzes wird über einen Zeichensatz-Browser ermöglicht, der auch frei skalierbare Zeichensätze, wie sie unter X11 R5 verfügbar sind, verwalten kann.



Malprogramm `xpaint`

Ghostscript/Ghostview

Ein wichtiger Bestandteil des Linux-Systems ist der Postscript-Interpreter *Ghostscript* (`gs`). Er erlaubt zusammen mit dem grafischen Frontend *Ghostview* die komfortable Anzeige von Postscript-Dateien. Doch Ghostscript realisiert nicht nur die Ausgabe auf den Bildschirm, sondern verwandelt auch einen gewöhnlichen Matrix- oder Tintenstahldrucker in ein Postscript-Ausgabegerät. Farbbilder werden bei der Wiedergabe auf nicht farbtauglichen Geräten mit Hilfe des Dithering-Verfahrens in vernünftiger Qualität präsentiert.



Postscript Interpreter `ghostview`

Momentan können nur Dateien ausgegeben werden, die dem Postscript-Level 1 entsprechen. Eine Level-2-Unterstützung ist jedoch schon angekündigt. Um auch Texte darstellen zu können, die einen der geschützten Adobe-Zeichensätze benutzen, wird zu Ghostscript ein Paket mit ähnlichen Zeichensätzen geliefert, die jedoch eine auffallend schlechte Qualität besitzen. Glücklicherweise arbeitet Ghostscript aber auch mit zusätzlich installierten Original-Schriften zusammen.

MPEG Video Player

Will man sich eines der zahlreichen MPEG-Videos aus dem Internet unter Linux ansehen, so kann dies über den MPEG-Player der Berkeley University erfolgen. Interessant ist, daß dazu keinerlei Spezialhardware benötigt wird und die Ausgabe auch über ein Netzwerk auf andere Bildschirme umgelenkt werden kann. Leider ist die Ablaufgeschwindigkeit sehr stark von der Leistungsfähigkeit des Rechners bzw. der Grafikkarte abhängig.

MPEG Video Player

11.4 Textbearbeitung

groff

Die klassische Art auf einem UNIX-System formatierte Dokumente zu erstellen, ist die Bearbeitung eines ASCII-Textes über das Kommando `nroff`. Hierbei handelt es sich um ein Paket zur Formatierung von Texten, Tabellen, Formeln und einfachen Bildern. Die Eingabe erfolgt im ASCII-Format und das Erscheinungsbild wird durch Angabe entsprechender Formatanweisungen beeinflußt. Die Ausgabe kann auf einem zeichenorientierten oder grafischen Gerät erfolgen. Alle UNIX Manual pages sind auf diese Weise erstellt worden. Die Anzeige von Manual pages dürfte normalerweise auch der Hauptverwendungszweck für diese Utilities sein. `nroff` kann durch externe Makros in seiner Funktionalität erweitert werden. Dazu existieren für unterschiedliche Ausgaben verschiedene Makro-Pakete. Auch zur Formatierung einer Manual page wird eine eigene Makro-Datei (`an`) benötigt.

```
linux2:/home> nroff -man ls.1 | more
```

Formatierung einer Manual page

Unter Linux steht jedoch nicht das Original `nroff`, sondern das erweiterte `groff`-Kommando zur Verfügung. Es wurde in einigen Punkten erweitert und erlaubt die direkte Ausgabe von Postscript- und TeX-DVI-Dateien.

TeX

Eine Welt für sich ist wohl das Satzsystem TeX (sprich tech oder teck) von Donald E. Knuth. Der Benutzer einer modernen WYSIWYG-Textverarbeitung wird sich im ersten Moment vielleicht in die Computer-Steinzeit zurückversetzt fühlen; doch TeX bietet einige Features, die es einer normalen Textverarbeitung in einigen Punkten überlegen machen. Ein Vorteil dieses Systems ist die kostenlose Verfügbarkeit auf beinahe allen Computerplattformen und die damit verbundene Portabilität der erzeugten Texte. TeX verarbeitet Dateien im ASCII-Format, die spezielle Formatieranweisungen enthalten. Besonders geeignet ist es zum Satz mathematischer Abhandlungen. Aber auch die Qualität normaler Texte übertrifft die mit einer normalen Textverarbeitung erstellten Dokumente. Die Eingabedatei wird von TeX in eine sogenannte DVI-Datei (device independent) übersetzt. Diese kann dann auf dem Bildschirm angezeigt oder auf einem Drucker ausgegeben werden. Das Format dieser DVI-Datei ist auf allen Computer-Systemen identisch, was einen völlig problemlosen Austausch erlaubt. Auch die direkte Ausgabe auf einem Satzbelichter ist kein Problem.

Previewer

Für TeX existieren zahlreiche Makro-Pakete und Zusatzprogramme, die auch unter Linux zur Verfügung stehen. Beispiele hierfür sind ein grafischer Previewer (`xdvi`), ein Utility zur Sortierung von Index-Dateien oder zur automatischen Erzeugung fehlender Zeichensätze. Außerdem gibt es zahlreiche Treiber, welche die von TeX erzeugten DVI-Dateien nach Postscript (`dvips`) konvertieren oder auf gewöhnlichen Druckern (Matrix, Tintenstrahl, Laser) ausgeben. Die Einbindung von Grafiken kann über LaTeX-Befehle oder aber über eine extern erstellte Postscript-Datei erfolgen.

LaTeX

Eines der bekanntesten Makro-Pakete ist LaTeX. Es stammt von Leslie Lamport und erleichtert den Umgang mit TeX erheblich. Natürlich existiert auch eine Variante für deutsche Texte. Mit Hilfe von LaTeX kann automatisch ein Inhaltsverzeichnis oder ein Index erstellt werden. Auch die Formatierung von Tabellen und Aufzählungen wird durch LaTeX wesentlich einfacher.

Im folgenden soll eine LaTeX-Datei und deren Ausgabe gezeigt werden.

```
\documentstyle[german]{article)
\topmargin -15mm
\headsep 0mm
\textwidth 16cm
\textheight 26cm
\oddsidemargin 0cm
\parindent 0mm

\begin{document}
\thispagestyle{empty}
\centerline{{\Huge \TeX, das Satzprogramm}}
\vspace{1cm}

\TeX\footnote{sprich "`tech"'} und das dazu verfügbare Makropaket
 \LaTeX\/ ermöglichen die Erstellung von Schriftstücken in
Satzqualität. Es eignet sich vor allem für Artikel, Bücher, Briefe,
mathematische Abhandlungen oder Dokumentationen. Gerade \LaTeX\/
bietet zahlreiche Gestaltungsmöglichkeiten, wie Formelsatz, Tabellen
oder Aufzählungen.  Inhaltsverzeichnisse und wissenschaftliche
Kapitelnumerierung werden automatisch erzeugt. Auch die Verwaltung
von Fußnoten macht keine Schwierigkeiten.
Zur Texthervorhebung stehen verschiedene Schriftarten zur Verfügung:

\begin{center}
{\rm Roman}, {\bf Bold Face}, {\tt Typewriter}, {\it Italic},
{\sl Slanted}, {\sc Small Caps}, {\sf Sans Serif}
\end{center}

Außerdem läßt sich die Textgröße verändern:

\begin{center}
{\tiny winzig}, {\scriptsize sehr klein}, {\footnotesize kleiner},
{\small klein}, {\normalsize normal}, {\large groß}, {\Large größer}\\
{\LARGE noch größer}, {\huge riesig}, {\Huge gigantisch}
\end{center}

Mathematische Formeln können wie folgt aussehen:

\begin{displaymath}
\int_0^\infty f(x)\,dx \approx \sum_{i=1}^n w_i e^{x_i} g(x_i)
\end{displaymath}
\begin{displaymath}
\sqrt[n]{\frac{x^n - y^n}{1 + u^{2n}}}
\end{displaymath}

Eine Aufzählung könnte zum Beispiel so aussehen:
\begin{itemize}
  \item Hardware
  \begin{itemize}
    \item Computer \item Tastatur \item Monitor
  \end{itemize}
  \item Software
  \begin{itemize}
    \item Betriebssystem
    \item Benutzeroberfläche
    \item Anwenderprogramm
  \end{itemize}
\end{itemize}

Tabellen lassen sich besonders einfach unter \LaTeX\/ setzten:

\begin{center}
\begin{tabular}{|r|l||c|rrr|c|c|} \hline
Platz & Verein & Sp. & S  & U  & N  & Tore  & Punkte\\ \hline\hline 1.
& Bayern München  & 33  & 19 & 13 & 1  & 66:31 & 51:15\\ \hline
2.    & Hamburger SV    & 33  & 18 & 9  & 6  & 65:37 & 45:21\\ \hline 3.
& Bor. M'Gladbach & 33  & 17 & 7  & 9  & 70:44 & 41:25\\ \hline 4.    &
Bor. Dortmund   & 33  & 14 & 10 & 9  & 66:50 & 38:28\\ \hline 5.    &
Werder Bremen   & 33  & 16 & 6  & 11 & 63:53 & 38:28\\ \hline 6.    &
Kaiserslautern  & 33  & 15 & 7  & 11 & 64:47 & 37:29\\ \hline
\end{tabular}
\end{center}

\newcommand{\absatz}{
\begin{minipage}[b]{7.5cm}
Ein Text kann auch in einer Text--Box umgebrochen werden. Übrigens
\TeX\/ verfügt selbstverständlich über eine automatische Silbentrennung.
Außerdem werden die einzelnen Buchstaben an bestimmten Stellen
untereinander geschoben. Dieser Vorgang wird {\em Kerning} genannt.
\end{minipage}}
```

```
\absatz
\hfill
\absatz


\begin{center}
{\Huge W\/elt V\/or V\/A W\/o}
\end{center}
\begin{center}
{\Huge Welt Vor VA Wo}
\end{center}
\end{document}
```

Beispiel für eine TeX Eingabedatei

Nach einem Übersetzungslauf kann das Ergebnis mit dem Kommando `xdvi` auf dem Bildschirm ausgegeben werden.

TEX, das Satzprogramm

TEX[1] und das dazu verfügbare Makropaket LATEX ermöglichen die Erstellung von Schriftstücken in Satzqualität. Es eignet sich vor allem für Artikel, Bücher, Briefe, mathematische Abhandlungen oder Dokumentationen. Gerade LATEX bietet zahlreiche Gestaltungsmöglichkeiten, wie Formelsatz, Tabellen oder Aufzählungen. Inhaltsverzeichnisse und wissenschaftliche Kapitelnumerierung werden automatisch erzeugt. Auch die Verwaltung von Fußnoten macht keine Schwierigkeiten. Zur Textihervorhebung stehen verschiedene Schriftarten zur Verfügung:

Roman, **Bold Face**, `Typewriter`, *Italic*, *Slanted*, SMALL CAPS, Sans Serif

Außerdem läßt sich die Textgröße verändern:

winzig, sehr klein, kleiner, klein, normal, groß, größer

noch größer, riesig, gigantisch

Mathematische Formeln können wie folgt aussehen:

$$\int_0^\infty f(x)\,dx \approx \sum_{i=1}^{n} w_i e^{x_i} g(x_i)$$

$$\sqrt{\frac{x^n - y^n}{1 + a^{2n}}}$$

Eine Aufzählung könnte zum Beispiel so aussehen:

- Hardware
  - Computer
  - Tastatur
  - Monitor
- Software
  - Betriebssystem
  - Benutzeroberfläche
  - Anwenderprogramm

Tabellen lassen sich besonders einfach unter LATEX setzten:

Platz	Verein	Sp.	S	U	N	Tore	Punkte
1.	Bayern München	33	19	13	1	66:31	51:15
2.	Hamburger SV	33	18	9	6	65:37	45:21
3.	Bor. M'Gladbach	33	17	7	9	70:44	41:25
4.	Bor. Dortmund	33	14	10	9	66:50	38:28
5.	Werder Bremen	33	16	6	11	63:53	38:28
6.	Kaiserslautern	33	15	7	11	64:47	37:29

Ein Text kann auch in einer Text-Box umgebrochen werden. Übrigens TEX verfügt selbstverständlich über eine automatische Silbentrennung. Außerdem werden die einzelnen Buchstaben an bestimmten Stellen untereinander geschoben. Dieser Vorgang wird *Kerning* genannt.

Ein Text kann auch in einer Text-Box umgebrochen werden. Übrigens TEX verfügt selbstverständlich über eine automatische Silbentrennung. Außerdem werden die einzelnen Buchstaben an bestimmten Stellen untereinander geschoben. Dieser Vorgang wird *Kerning* genannt.

Welt Vor VA Wo

Beispiel für TeX-Ausgabe

11.5 Spiele

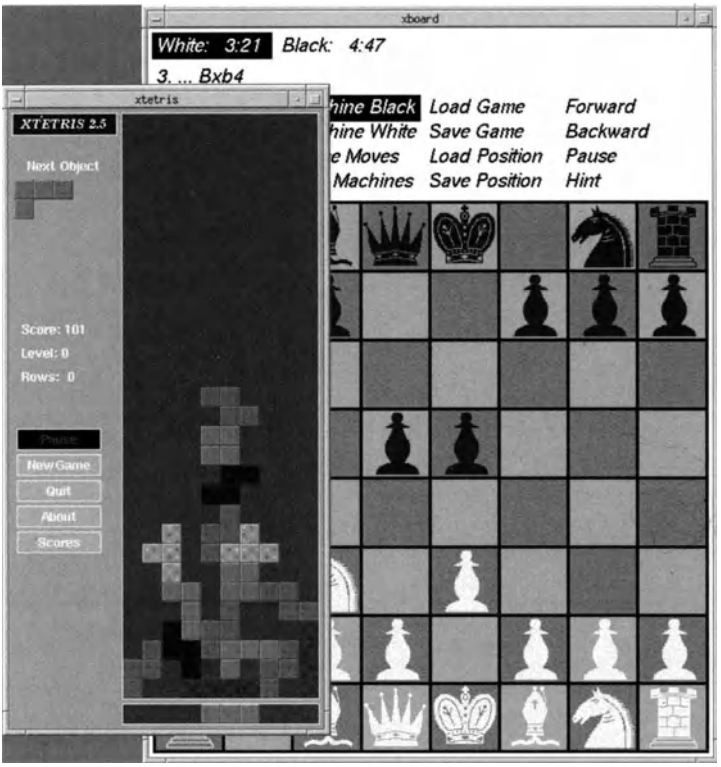
Will man sich nach getaner Arbeit unter Linux ein wenig entspannen, so kann man auf eines der vielen Spiele zurückgreifen.

Tetris

Ein sehr populäres Spiel, das auch unter Linux in einer ansprechenden Form zur Verfügung steht, ist Tetris. Der Spieler muß durch eine geschickte Verteilung der herunterfallenden Steine dafür sorgen, daß diese sich nicht zu hoch auftürmen. Dies wäre natürlich recht schnell der Fall, würden nicht vollständig aufgefüllte Zeilen automatisch verschwinden.

GNU-Schach

Sucht man etwas anspruchsvollere Ablenkung, so sollte man sich `xboard` und GNU-Chess näher ansehen. Die spielerischen Fähigkeiten dieses Schachprogrammes sind recht beachtlich.



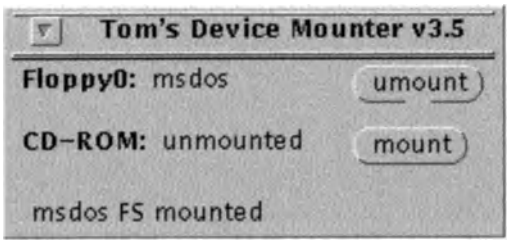
xboard und xtetris

Es hat auf einschlägigen Wettbewerben auch schon kommerzielle Schachprogramme geschlagen. GNU-Chess selbst ist nicht gerade benutzerfreundlich.
Es bedarf vielmehr eines grafischen Frontends (`xboard`), um es vernünftig spielen zu können. `xboard` kann GNU-Chess auch zweimal starten und gegeneinander spielen lassen. Eine Option erlaubt auch Partien im Internet. So findet man wohl zu jeder Tages- und Nachtzeit einen adäquaten Partner irgendwo in der Welt.

11.6 Sonstiges

XVMount

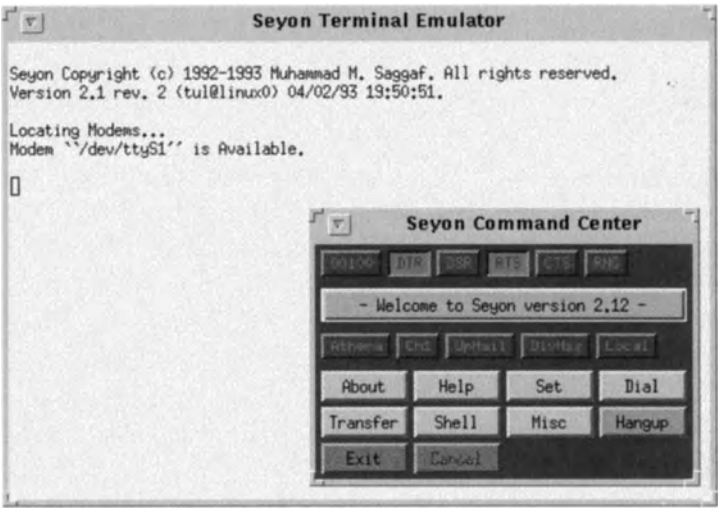
Ein paraktisches Utility, das den täglichen Umgang mit Speichermedien wie Disketten, Wechselplatten oder CD-ROMs erleichtert, ist `xvmount`. Will man unter UNIX ein neues Dateisystem ansprechen, so muß es bekanntlich vorher über den Befehl `mount` in das Filesystem eingebunden werden. Dabei handelt es sich jedoch um ein privilegiertes Kommando, das nur vom Superuser ausgeführt werden darf. `xvmount` erlaubt auch dem normalen Anwender, bestimmte Filesysteme zu mounten. Normalerweise muß beim Mounten auch der Typ des Filesystems angegeben werden. `xvmount` dagegen erkennt diesen selbständig, so daß sich der Anwender auch darum nicht kümmern muß. Die Bedienung wird durch eine benutzerfreundliche grafische Oberfläche sehr einfach.



xvmount Utility

Seyon

Zur Datenübertragung und zum Einloggen in ein anderes System über ein Modem wird ein Terminalprogramm benötigt. Seyon ist ein solches. Als Terminalemulator wird `xterm` herangezogen, so daß eine vollwertige VT100 Terminal-Emulation zur Verfügung steht. Da das Programm eine X11-Benutzeroberfläche besitzt, lassen sich die Schnittstellenparameter komfortabel mit der Maus einstellen. Neben einem Nummernverzeichnis verfügt das Programm über einen einfachen Kommandointerpreter.
Beim Empfang von Dateien über das Zmodem-Protokoll wird das entsprechende (`rz`) Empfangsprogramm automatisch gestartet. Optional können alle Bildschirmausgaben in einer Datei mitprotokolliert werden.

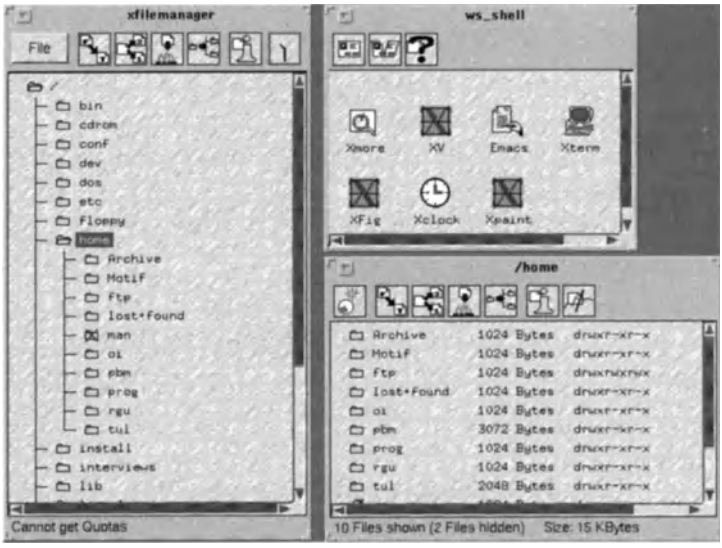


Terminalprogramm `seyon`

X-Filemanager

Vielen UNIX-Anwendern ist die Kommandozeilenumgebung der üblichen Shells zur Dateiverwaltung nicht ausreichend. Sie bevorzugen, trotz des großen Leistungsumfangs der UNIX-Shells, grafische Oberflächen. Ein relativ neues Programm zur grafischen Darstellung von Dateibäumen und zum Kopieren,

Umbenennen und Verschieben von Dateien ist der X-Filemanager. Der Benutzer kann sich in mehreren Fenstern einen Überblick über seine Dateistruktur verschaffen und über die Maus Dateien und Verzeichnisse kopieren. In einem separaten Fenster können Programme abgelegt werden, die besonders häufig benötigt werden. Auf Wunsch kann bei einem Doppelklick auf eine Datei ein passendes Programm zur Bearbeitung oder zur Ausgabe auf den Bildschirm gestartet werden. Die Darstellung der einzelnen Dateien und Verzeichnisse erfolgt durch kleine Icons, die auch farbig sein können. Für den Benutzer ist es möglich, die Zuordnung zwischen den Datei-Typen und Icons individuell festzulegen.



Datei Manager `xfilemgr`

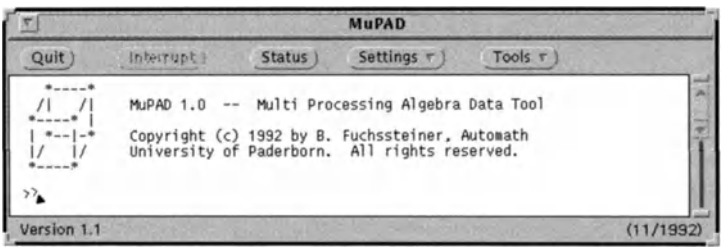
Ingres und Postgres

Wie schon erwähnt, sind auch die Datenbanksysteme Ingres und der Nachfolger Postgres der Berkeley University für Linux portiert worden. Diese Version von Ingres sollte jedoch nicht mit dem kommerziell vertriebenen SQL-Datenbanksystem verwechselt werden. Sie ist wesentlich älter und besitzt anstelle von SQL eine etwas einfachere Abfragesprache.

Postgres ist eine objektorientierte Datenbank, die jedoch nicht wie viele der kommerziellen Systeme auf C++ basiert, sondern einen etwas eigenwilligen Ansatz verfolgt.
Beide Systeme sind für Studienzwecke recht interessant, für die Anwendungsentwicklung jedoch nur bedingt verwendbar.

MuPAD

Ein für Studenten sehr interessantes Programmpaket ist *MuPAD*. Dieses Computer-Algebra-System wurde an der Universität Paderborn entwickelt und wird kostenlos an nicht kommerzielle Institutionen abgegeben. Es erlaubt die Bearbeitung umfangreicher mathematischer Problemstellungen. Eine integrierte Programmiersprache ermöglicht die Implementierung eigener Algorithmen. Besonderes interessant ist die Tatsache, daß MuPAD vor allem zur Verarbeitung paralleler Probleme konzipiert wurde.
Zukünftige Versionen werden daher sehr effizienten Gebrauch von Multiprozessor-Architekturen oder Wortstation Clustern machen können. Zur Fehlersuche steht ein eigener Debugger mit grafischem Frontend zur Verfügung.
MuPAD verfügt zunächst über eine einfache Kommandozeilen-orientierte Benutzerschnittstelle. Ein eigenes OpenLook-basiertes Frontend erleichtert den Umgang jedoch erheblich. Recht ausgereift sind die grafischen Ausgabefunktionen, die z.B. die Darstellung dreidimensionaler Funktionen ermöglichen.
Eine Hypertext Online-Hilfe ermöglicht den Zugriff auf alle wesentlichen Infomationen und Funktionsbeschreibungen. Ein ausführliches Handbuch zu MuPAD ist im Birkhäuser Verlag Basel erschienen.



Computer-Algebra-System MuPAD

Netzanwendungen

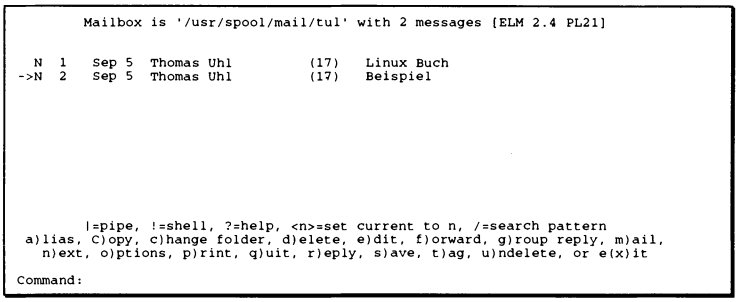
Die effektive Entwicklung und schnelle Verbreitung von Linux ist ein Phänomen, das ohne die vielen Kommunikationsmöglichkeiten und Dienste innerhalb des Internets nicht erklärbar ist.
Da viele dieser Möglichkeiten außerhalb der Hochschulen nicht bekannt sind, soll hier ein grober Überblick über die wichtigsten dieser Dienste gegeben werden.

12.1 Mail

Das Electronic Mail System unter UNIX besteht meist aus mehreren Teilen: einem Frontend, mit dem Briefe ("Mails") gelesen und geschrieben werden können und einem Daemon (`smail` oder `sendmail`), der im Hintergrund läuft und für das Zustellen der Mails zuständig ist. Ein normaler Benutzer kommt mit diesem Daemon nicht in Berührung und benutzt im Normalfall ein passendes Frontend.
Ein weit verbreitetes Programm zum Lesen und Schreiben von E-Mail ist `elm`. Es ist in fast allen Linux-Distributionen enthalten und existiert auch für die meisten anderen UNIX-Plattformen.

elm

Beim ersten Starten von `elm` werden zwei Unterverzeichnisse im Home-Verzeichnis des Benutzers angelegt. Ein Konfigurationsverzeichnis und ein Verzeichnis, in dem empfangene Mails abgelegt werden.



Das Mail-Frontend elm

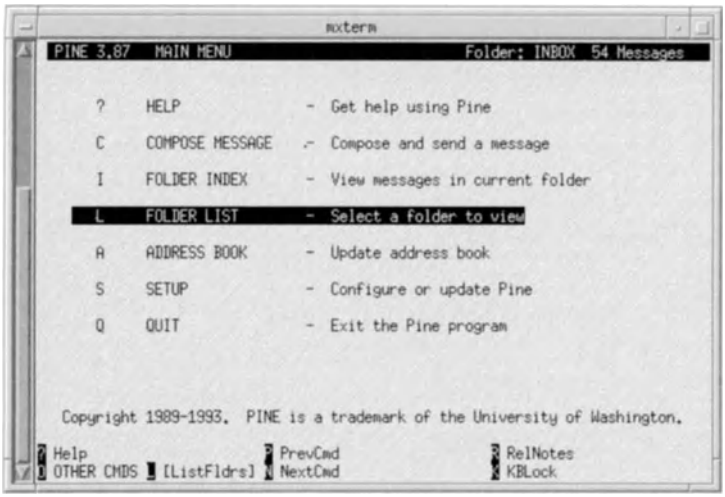
Neue Nachrichten werden in einer Liste mit dem Absender und dem Titel der Nachricht angezeigt. Sie können mit den Pfeiltasten ausgewählt, gelesen und in verschiedene Dateien abgelegt werden. Für das Schreiben von Mails wird ein externer Editor aufgerufen, den man in den `elm` Optionen definieren kann.

MIME

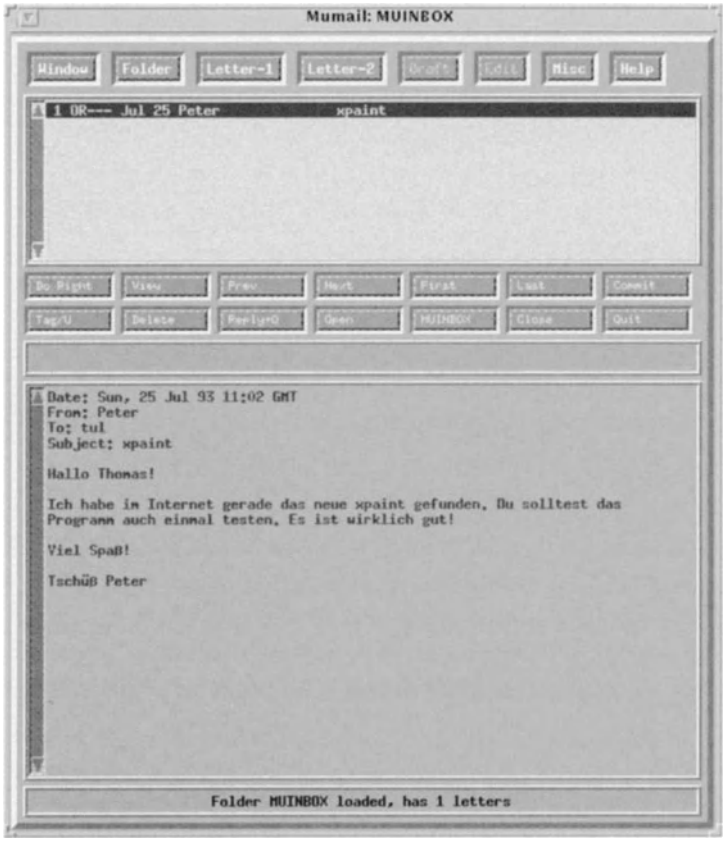
Nachrichten im *Multipurpose Internet Mail Extension* Format (MIME), die beispielsweise Bilder, Sounds oder Programme enthalten können, werden ebenfalls erkannt. Sie können mit dem `metamail`-Programm und den jeweiligen Präsentationsprogrammen ausgegeben werden.

pine, pico, IMAP

Mehr Komfort als `elm` bietet `pine`. `Pine` steht für *Pine Is No-longer Elm* oder *Program for Internet News & Email.* Einer der wesentlichen Unterschiede zu `elm` ist die Möglichkeit, auch auf Internet-News (siehe nächsten Abschnitt) zugreifen zu können. `Pine` enthält einen eigenen Editor namens `Pico`, der die Erstellung von Mails wesentlich erleichtert. Da `pine` das IMAP2-Protokoll benutzt, können auch Mailboxen verwaltet werden, die sich nicht auf der lokalen Maschine befinden. Dies ist vor allem dann interessant, wenn man oft zwischen verschiedenen Maschinen wechselt. IMAP2 gewährleistet einen konsistenten Zugriff auf die Mails des "Heimatrechners" und unterstützt mittlerweile auch Nachrichten nach dem MIME-Standard.



Das Mailfrontend `pine`



Grafisches Mailfrontend MuMail

Ein Mail-Programm mit grafischer Oberfläche ist `MuMail`. Es verwendet X11 und läßt sich komfortabel mit der Maus bedienen.

MuMail

Der Funtionsumfang entspricht ungefähr dem von `elm`. Auch damit können Multimedia-Mails erzeugt und angezeigt werden.

12.2 News

News ist eines der wichtigsten Medien, um Information über neue Entwicklungen im Internet zu bekommen.

Struktur und Aufbau

Das Prinzip ist recht einfach. Es gibt an sehr vielen Stellen im Internet, beispielsweise an fast jeder Hochschule, sogenannte Newsserver. Auf einem Newsserver werden verschiedene Newsgruppen verwaltet, die man sich wie schwarze Bretter zu einem bestimmten Thema vorstellen kann. Diese Themen reichen von den verschiedenen Betriebssystemen und Programmen über Freizeitbeschäftigungen, Sport oder Computerspiele bis hin zu sozialen Themen. Insgesamt existieren mehrere Tausend solcher Newsgruppen weltweit.

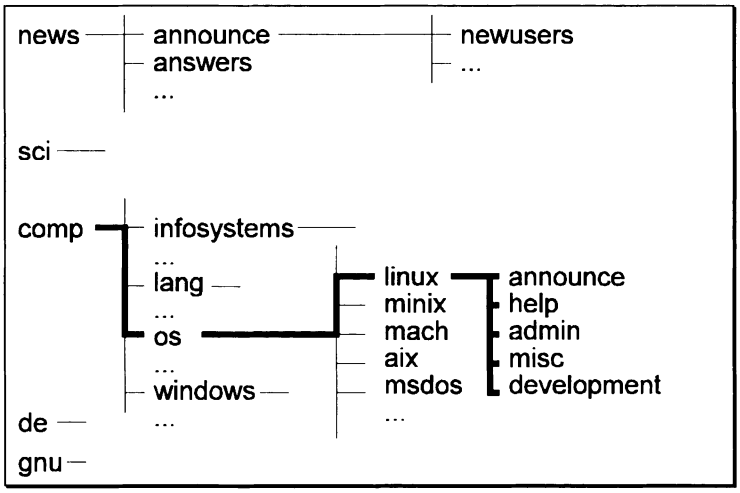
Alle Newsserver tauschen ständig ihre Nachrichten mit den benachbarten Newsservern aus, so daß eine Nachricht, die auf einem Newsserver in eine bestimmte Gruppe geschickt wurde, sich recht schnell auf alle Newsserver verbreitet. Diese Nachrichten werden auf den Newsservern eine bestimmte Zeit, zum Beispiel zwei Wochen, gehalten und danach gelöscht.

Die Gesamtheit aller Rechner, die Nachrichten in Form von News austauschen, wird als USENET bezeichnet. Die Kommunikation zwischen diesen Rechnern basierte ursprünglich auf Modems und `uucp` (UNIX to UNIX copy) und war daher wesentlich langsamer als die Verbindungen im Internet, die inzwischen anstelle von uucp ein spezielles Protokoll mit dem Namen NNTP verwenden.

USENET

Newsgruppen

Die Namen der Newsgruppen sind hierarchisch aufgebaut und bestehen aus Teilnamen, die jeweils durch Punkte getrennt sind. `comp.os.linux.announce` bedeutet zum Beispiel, daß diese Newsgruppe in der Hierarchie `comp` existiert, sich also mit Computern, Software oder Informatik im allgemeinen beschäftigt. `os` steht für Operating Systems und `comp.os.linux` ist also der Beginn der Namen aller Newsgruppen zu Linux, die sich unterhalb der `comp`-Hierarchie befinden.



kleiner Ausschnitt aus der Hierarchie der Newsgruppen

Neben `comp` gibt es mehrere weitere Haupthierarchien wie `sci` für Gruppen, die sich hauptsächlich mit wissenschaftlichen Themen beschäftigen oder `de` für deutschsprachige Gruppen.

Moderierte Gruppen

Gruppen können entweder offen für jedermann oder moderiert sein. Moderiert bedeutet, daß es einen Moderator gibt, der entscheidet, welche Nachrichten für die Allgemeinheit interessant sind und erscheinen, und welche nicht. Bei Linux ist dies zum Beispiel die Gruppe `comp.os.linux.announce`, in der nur Nachrichten über Ankündigungen neuer Programme oder

Systemerweiterungen veröffentlicht werden. Dies ist vor allem dann nötig, wenn die Anzahl der Nachrichten in den anderen Gruppen so groß ist, daß man nur noch mit einem hohen Zeitaufwand auf dem Laufenden bleiben kann.
Unmoderierte Gruppen sind offen, so daß jeder Nachrichten an die Gruppe schreiben kann.

Regeln und Nettiquette

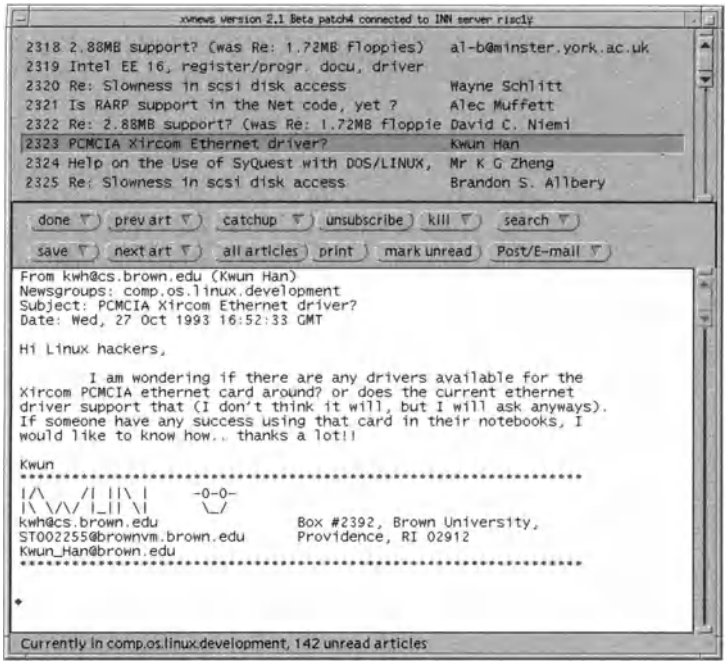
Eine neue Gruppen wird erstellt, wenn dies von einem USNET-Teilnehmer vorgeschlagen wird, und der Vorschlag in einer Abstimmung angenommen wurde. Die Diskussionen über solche neuen Newsgruppen finden in der Regel in der Gruppe `news.groups` statt.
Beim Senden von Nachrichten an eine Newsgruppe sollte man bestimmte Regeln beachten, die als "Nettiquette" bezeichnet werden. Beleidigende Kommentare oder persönliche Angriffe werden nicht geduldet. Eine Übersicht über die Verhaltensregeln findet man in der Newsgruppe `news.announce.newusers`.

Newsreader

Um News lesen zu können, benötigt man einen Newsreader und einen über ein Netzwerk oder Modem erreichbaren Newsserver. An Universitäten ist dies in der Regel kein Problem, da diese meist einen eigenen Newsserver betreiben.
Newsreader gibt es inzwischen für fast alle Rechner und Betriebssysteme, sofern sie ein Netzwerk unterstützen. Bekannte Newsreader sind zum Beispiel `rn`, `trn`, `tin`, `xrn` oder `xvnews` für X-Windows oder `trumpet` für PCs.

tin, xrn, xvnews

Unter Linux existieren fast alle dieser Newsreader. Da XView und der Windowmanager `olvwm` inzwischen einen gewissen Standard unter Linux darstellen und auch in den gängigen Installationspaketen enthalten sind, bietet sich der XView-basierte `xvnews` an. Die einzige Konfiguration, die für diesen Newsreader benötigt wird, ist eine Environmentvariable mit dem

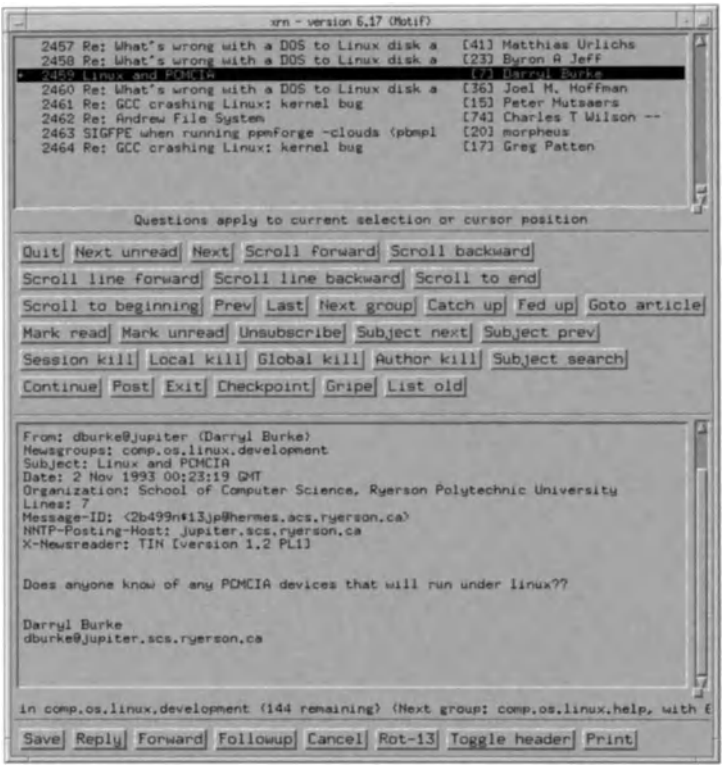


News Reader `xvnews`

Namen `NNTPSERVER`, die den Hostnamen des Newsservers enthalten muß.

Recht brauchbar ist auch `xrn`, der sowohl in einer Athena-Widget- als auch einer Motif-Variante erhältlich ist.

xrn

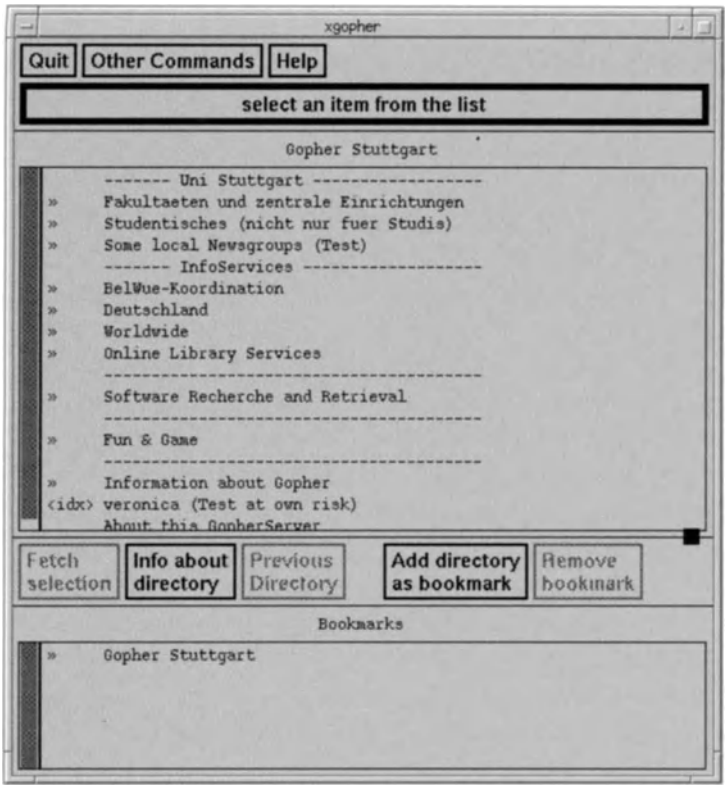


xrn unter OSF/Motif

12.3 Gopher

Gopher ist ein Internet Service, der viele der Dienste, die von einzelnen Universitäten oder anderen Einrichtungen weltweit angeboten werden, unter einer gemeinsamen Oberfläche verfügbar macht.

Für den Anwender stellt sich Gopher wie ein einziges, großes hierarchisches Menü dar. Innerhalb dieser Stuktur kann man sich auch von einem zum nächsten Server bewegen und auf die verschiedensten Dienste zugreifen. Angeboten werden zum Beispiel Literaturdatenbanken, Wetterdienste, Dokumente aller Art (Text, Bilder, Ton) Mensaspeisepläne einiger Universitäten, Telnet-Sessions und Gateways zu anderen Diensten wie WAIS.



Gopher Client `xgopher`

Immer mehr Einrichtungen bieten auch Information, die bisher nur gedruckt verfügbar war, in ihrem Gopher Service an.

XGopher

Um Gopher zu benutzen, benötigt man einen Gopher-Client. Unter Linux bietet sich hierfür der unter X-Windows ablaufende `xgopher` an. Beim Start stellt er eine Verbindung zu einem eingestellten Gopher-Server her und man kann sich dann frei im Internet bewegen.

12.4 IRC

Eine Möglichkeit, sich direkt mit mehreren Anwendern im Internet zu unterhalten, bietet IRC. IRC steht für Internet Relay Chat und ist, ähnlich wie News, ein System von vernetzten Servern, die Information untereinander austauschen. Wie bei

News gibt es eine Gliederung nach Themen, die hier Channel (Kanal) genannt wird.
Der große Unterschied zu News ist, daß der Austausch ohne Verzögerung vor sich geht und man sich direkt mit den anderen Teilnehmern unterhalten kann. Dies ist vor allem dann interessant, wenn man ein akutes Problem hat, das man sofort mit anderen besprechen möchte.
Der Zugang erfolgt mit einem speziellen `irc`-Client-Programm, das die Verbindung zum nächsten IRC-Server herstellt. Innerhalb des IRC-Clients gibt es bestimmte Befehle, die alle mit einem / beginnen und es zum Beispiel ermöglichen, einen Channel zu betreten oder zu verlassen. Text, den man ohne Befehl eingibt, wird sofort übertragen und erscheint bei allen anderen IRC-Teilnehmern, die in diesem Moment im gleichen Channel sind.
Auch in IRC gibt es einen Channel, in dem über Linux diskutiert wird. Um sich daran zu beteiligen startet man das IRC-Programm und gibt dann `/join #linux` ein. `/join` ist der Befehl um einen Channel zu betreten.
Danach erscheint jede Zeile, die in diesem Channel abgeschickt wird, mit dem vorangestellten Namen des Absenders.

Kommunikation über IRC

Die Anzahl der Channels ändert sich ständig, da jeder Benutzer eigene Kanäle eröffnen kann. Dies geschieht ebenfalls mittels

`/join`. Ist der übergebene Channel noch nicht vorhanden, so wird er neu angelegt. In der Regel sind mehrere hundert Kanäle gleichzeitig aktiv, die man sich mit `/list` ausgeben lassen kann. Neben der Möglichkeit, Online-Diskussionen durchzuführen, sieht das IRC-Protokoll auch den Austausch von kleineren Dateien vor. Außerdem lassen sich auch private Mitteilungen übertragen.

12.5 Archie

Bei der großen Anzahl von FTP-Servern und der darauf verfügbaren Free Software, ist es meist nicht einfach, im Bedarfsfall die geeignete Software zu finden. Eine wertvolle Hilfe leisten hier sogenannte Archie-Server.
Ein Archie Server erlaubt den Zugriff auf eine mehrere Gigabyte große Datenbank, die die Inhaltsverzeichnisse der wichtigsten FTP-Server im Internet enthält. Diese Datenbank wird automatisch in bestimmten Zeitintervallen auf den neuesten Stand gebracht.
Einen Archie-Server findet man unter folgenden Adressen:

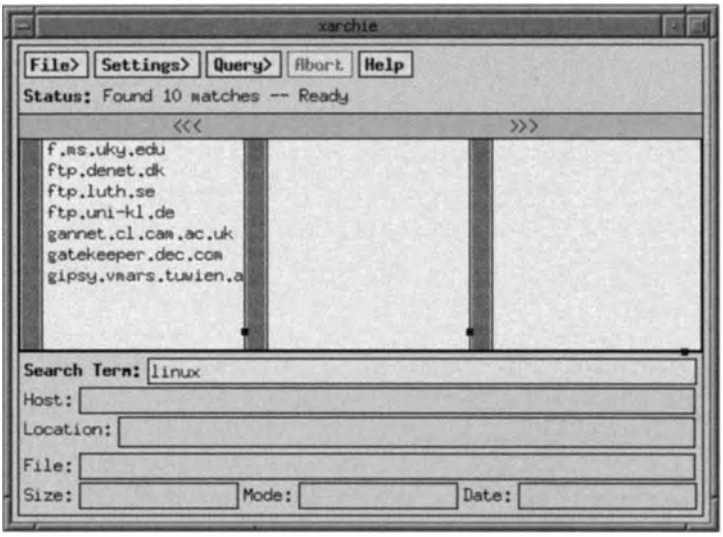
- `archie.th-darmstadt.de` (Deutschland)
- `archie.funet.fi` (Finnland)
- `archie.ans.net` (New York)

Um einen Server abzufragen, kann man sich mit `telnet` und dem Usernamen `archie` auf dem entsprechenden Host einloggen und mit einer einfachen Abfragesprache nach Programmen suchen.
Weit komfortabler geht dies jedoch mit dem grafischen Frontend `xarchie`, welches ebenfalls unter Linux verfügbar ist. Hier gibt man nur den Suchbegriff ein, wählt einen Suchmodus und `xarchie` stellt die Verbindung zum eingestellten Server her. Nach kurzer Zeit werden die gefundenen Dateien und deren Server in einem Browser dargestellt.
Mit neueren Versionen von `xarchie` kann dann sogar direkt eine FTP-Übertragung gestartet werden und die gewünschte Datei direkt vom ausgewählten FTP-Server geladen werden.

xarchie

whatis

Neben der Suche nach Programmen mit einem bestimmten Namen bietet ein Archie-Server auch eine sogenannte `whatis` Datenbank an, in der mit dem Kommando `whatis` eine kurze Beschreibung zu Programmen abgefragt werden kann. Wenn man beispielsweise den Namen einer gesuchten Datei überhaupt nicht oder nur Bruchteilhaft kennt, so kann man sich mit `whatis` eine Liste aller registrierten Programme mit ihrer Beschreibung und deren Namen ausgeben lassen.



Dateisuche über `xarchie`

Wer keinen direkten Internet-Zugang besitzt, kann dennoch die Archie-Server per E-Mail abfragen. Dazu sendet man bestimmte Befehle an den Benutzer archie auf einem solchen Server. Eine genauere Beschreibung erhält man wie bei FTP-Mailservern durch Senden des Befehls `help` in der ersten Zeile der Mail.

12.6 WWW und Mosaic

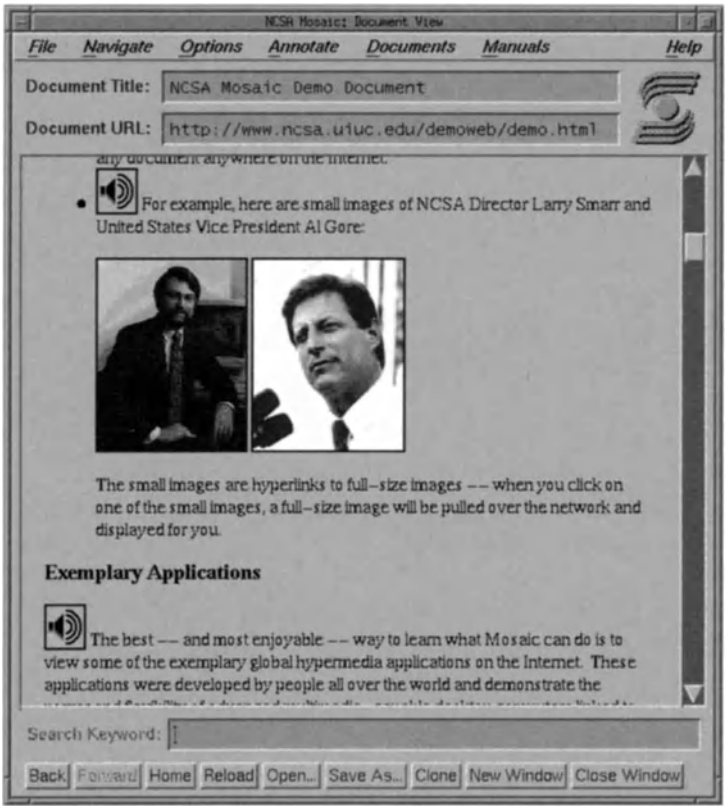
Mosaic ist ein sehr komfortables Programm, mit dem auf viele der oben genannten Internet-Dienste von einer gemeinsamen Benutzeroberfläche zugegriffen werden kann. Unterstützt werden

Internet-Dienste wie Gopher, News, World Wide Web (WWW), WAIS, Archie und FTP.

WWW ist ein weiterer Internet Service, bei dem auf verteilte Multimedia-Hypertext-Dokumente zugegriffen wird, die neben formatierten Texten auch Verweise auf andere Dokumente oder auch Bilder, Videos oder Audio-Dateien enthalten können. Die Beschreibung dieser Dokumente erfolgt über eine sogenannte Hypertext Markup Language (HTML) im ASCII-Format.

WWW

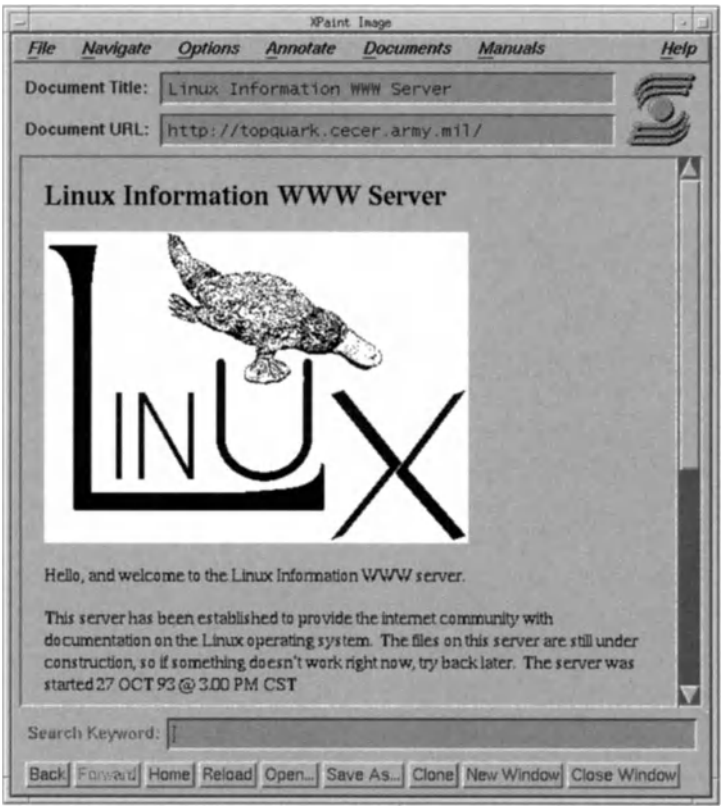
Um dieses Programm zu übersetzen wird OSF/Motif benötigt, das nicht kostenlos erhältlich ist. Es gibt jedoch auch eine fertig übersetzte Version dieses Programms, die statisch gelinkt ist und so kein OSF/Motif-Laufzeitsystem benötigt.



WWW-Zugriff über Mosaic

HTTP Server

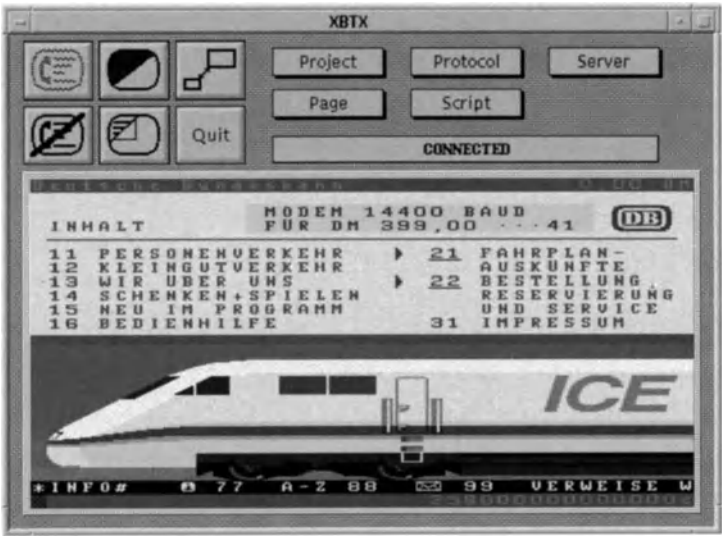
Mit Hilfe eines HTTP-Servers ist es auch unter Linux problemlos möglich, einen eigenen, verteilten Infomationsdienst aufzubauen. Ein derartiger Server, der als Daemon im Hintergrund läuft, wartet auf eine Socket-Verbindung über einen definierten Port und stellt nach erfolgreicher Kontaktaufnahme mit einem WWW-Client (z.B. Mosaic) die Dateien eines bestimmten Unterverzeichnisses zur Verfügung. Falls ein direkter Internetzugriff möglich ist, können in den lokalen Dokumenten auch entsprechende Links auf andere HTTP-Server angeboten werden. Ein vor allem für Linux-Anwender interessanter Server ist beispielsweise `http://topquark.cecer.army.mil`. Dort findet man unter anderem eine Zusammenfassung der wichtigsten Linux-Dokumente, wie FAQs oder HOW-TOs.



HTTP-Server mit aktuellen Infos über Linux

12.7 XBTX

Wer einen Zugang zum Bildschirmtext bzw. Datex-J der Telekom besitzt, kann diesen auch unter Linux nutzen. Neben dem Modem wird das XBTX-Paket (Universität Erlangen) benötigt. Es besteht aufgrund seiner Client/Server-Architektur aus zwei Komponenten. Der BTX-Server kontrolliert die Modem-Verbinung und leitet die empfamgenen Daten an den BTX-Client weiter, der natürlich auch auf einer anderen Maschine laufen kann.



BTX (Datex-J) unter Linux

Anhang

13.1 Übersicht /etc-Dateien

Die meisten Konfigurationsdateien liegen im Verzeichnis `/etc` oder sie sind aus dem Unterverzeichnis `/conf` gelinkt. Im folgenden soll eine Übersicht über die wichtigsten dieser Dateien gegeben werden. Der genaue Aufbau der Dateien wird dabei nicht erläutert.

- **`clock*`** - Programm zum Abfragen und Setzen der CMOS Uhr
- **`crond*`** - Cron-Daemon. Startet Programme zu bestimmten Zeiten. Die Programme, die gestartet werden, können mit dem Befehl crontab an den Cron-Daemon weitergeleitet werden.
- **`domain`** - Datei, die den Namen der eigenen Internet Domain enthält.
- **`exports`** - Datei, in der die per NFS exportierten Verrzeichnisse definiert werden.
- **`fdprm`** - Parameter für Diskettenlaufwerke
- **`fingerd*`** - Finger Daemon
- **`fstab`** - In dieser Datei stehen die Filesysteme und die Swap-Partitions, die beim Booten gemounted bzw. aktiviert werden.
- **`ftpd*`** - FTP-Daemon
- **`ftpaccess`** - Konfigurationsdatei für den FTP-Daemon. Hier können Zugangsbeschränkungen und Nachrichten definiert werden.

- **ftpusers** - Konfigurationsdatei für den FTP-Daemon. Benutzer, die in dieser Datei stehen, können sich nicht per FTP einloggen.
- **getty, uugetty** - Programm, das das Einloggen ermöglicht.
- **gettydefs, gettytab** - Konfigurationsfiles für getty
- **group** - In dieser Datei stehen die Gruppen mit allen Benutzern, die zu der Gruppe gehören. Ist ein Benutzer mehreren Gruppen zugeordnet, so wird er in dieser Datei bei mehreren Gruppen angeben. In der Datei passwd ist nur seine primäre Gruppe angegeben.
- **groupadd*, groupdel*, groupmod*** - Programme zum Verwalten von Gruppen
- **gshadow** - Shadow-Datei für die Datei `group`
- **halt** - Programm zum Beenden von Linux
- **host** - Enthält den eigenen Hostnamen
- **host.conf** - In dieser Datei wird die Reihenfolge der Suche nach einer IP-Adresse angegeben. `order hosts bind` bedeutet zum Beispiel, daß zunächst die Datei hosts durchsucht wird, und falls dort der gesuchte Host nicht eingetragen ist, der Nameserver befragt werden soll.
- **hostcvt.build*** - Script zum Erstellen der Konfigurationsdateien für einen Nameserver aus der Datei `hosts`
- **hosts** - Datei mit den Namen und IP-Adressen von anderen Rechnern. Diese Datei wird je nach Einstellung in der Datei `host.conf` vor oder nach einem Nameserver durchsucht, um die IP Adresse eines Hosts zu finden.
- **inetd*** - Daemon, der die meisten anderen Internet-Daemons kontrolliert. Er wartet auf TCP/IP-Verbindungen und startet entsprechend dem Port der Verbindung den zugehörigen Daemon.

TCP/IP

- **inetd.conf** - Konfigurationsdatei für den Daemon `inetd`. Hier wird angegeben, welcher Daemon bei einer Verbindung auf einem bestimmten Port gestartet werden soll.
- **init** - Programm, das als erstes gestartet wird, den Bootvorgang und die System Runlevels kontrolliert und dafür sorgt, daß keine "zombie" Prozesse entstehen.

- **inittab** - Konfigurationsdatei für init. Hier werden Programme bestimmten Runlevels zugeordnet.
- **issue** - Textdatei, die vor dem Login-Prompt ausgegeben wird. Normalerweise steht in dieser Datei der Name des Hosts und ein Begrüßungstext.
- **login.defs** - Konfigurationsdatei für `login`. Hier wird unter anderem eingestellt, von welchen Geräten ein Login als roor möglich ist und welche Textdateien nach dem Einloggen ausgedruckt werden
- **logoutd*** - Daemon, der Benutzer automatisch ausloggt.
- **lpc*** - Programm zum Ändern von Eintragungen und Optionen der Drucker und Warteschlangen. Drucker
- **lpd*** - Drucker-Daemon. Verwaltet die Druckerwarteschlangen und die Verbindung mit Remote-Druckern.
- **lpr*** - Programm zum Drucken
- **magic** - Datei, die eine Zuordnung von Bytemustern, die am Anfang einer Datei stehen, zu Dateitypen herstellt. Durch Suchen in dieser Datei kann der Typ einer anderen Datei festgestellt werden.
- **mkpasswd*** - Programm zum Erstellen der Hilfsdateien für schnelleren Zugriff auf die Dateien `passwd` und `group`.
- **motd** - Textfile (message of the day), das normalerweise automatisch nach dem Einloggen angezeigt wird.
- **mount*** - Programm zum Mounten von Filesystemen.
- **mountd*** - Mount-Daemon. Ist für NFS-Mounts zuständig.
- **mtab** - Interne Tabelle für mount. In ihr sind die aktuell gemounteten Filesysteme eingetragen.
- **mtools** - Konfiguratiosdatei für die mtools.
- **named*** - Nameserver-Daemon Nameserver
- **named.boot, named.boot.in, named.hosts, named.local, named.rev, netlist, netlist.in, soabasefile, soabasefile.in, softland** - Konfigurationsdateien für den Nameserver bzw. das `hostcvt.build`-Script im SLS-Paket
- **named.off*, named.reload*, named.restart*, named-xfer*, nextserial*** - Hilfsprogramme für den Nameserver

- **netdate*** - Programm zum Übertragen der Zeit von einem anderen Host
- **newusers*, chpasswd*** - Programme zum schnellen Anlegen / Ändern von vielen Benutzern und Passwörtern
- **nfsd*** - NFS-Daemon
- **ntalkd*** - Talk-Daemon
- **pac*** - Programm für die Druckerverwaltung und Abrechnung
- **passwd** - Datei, in der die Benutzer definiert werden. Hier werden die User Ids, Benutzernamen, Login-Shells und die primären Gruppen definiert. die eigentlichen Passwörter stehen in der Datei `shadow`, die im Gegensatz zur `passwd`-Datei nur mit root-Privilegien gelesen werden kann.
- **portmap*** - Daemon für die Verwaltung von RPC-Programmen.
- **printcap** - Datei, in der die Drucker definiert werden.
- **protocols** - In dieser Datei sind die TCP/IP-Pakettypen definiert.
- **pwconv*, pwunconv*** - Programm zum Konvertieren zwischen einer normalen `passwd`-Datei und den Shadow-Dateien.
- **rc** - Script, das beim Systemstart von Init als erstes ausgeführt wird.
- **rc.local** - Dieses Script wird beim Systemstart von dem Script rc ausgeführt.
- **rc.net** - Script, das beim Systemstart von dem Script rc.local gestartet wird. Hier werden die Netzadapter konfiguriert und die Netzwerk Daemons gestartet.
- **reboot** - Programm zum herunterfahren und Neustarten des Rechners.
- **resolv.conf** - Konfigurationsdatei für das TCP/IP-System. Hier wird der eigene Domainname und die Adresse des Nameservers eingetragen.
- **rlogind*** - Daemon für rlogin
- **rpc** - Konfigurationsdatei für RPC-Programme.
- **rpcinfo*** - Programm zum Testen und Abfragen des rpc Systems
- **rshd*** - `rsh`-Daemon

- **services** - Datei, in der die TCP/IP-Ports den Services (Programmen) zugeordnet werden.
- **shadow** - Datei mit den verschlüsselten Passwörtern der Benutzer
- **shells** - In dieser Datei werden die gültigen Shells des Systems eingetragen. Der FTP-Daemon überprüft beim Einloggen eines Benutzers mit ftp, ob die Login-Shell des Benutzers in dieser Datei eingetragen ist. Wenn nicht, wird das Einloggen verweigert.
- **shutdown** - Programm zum Beenden des Systems
- **swapon*, swapoff*** - Programme zum aktivieren bzw. deaktivieren der Swap-Files oder Partitionen.
- **syslogd*, syslogk*** - Syslog Daemons
- **syslog.conf** - Konfigurationsfile für den Syslog Daemon.
- **syslogd.reload*** - Hilfsprogramm, um den Syslog Daemon erneut zu starten
- **sysperms, syssetup** - Hilfsscripts, die beim Installieren des Systems verwendet werden.
- **telnetd*** - `telnet` Daemon
- **termcap** - Konfigurationsdatei für die `termcap` Library. Hier werden die Funktionen der verschiedenen Terminaltypen definiert.
- **tftpd*** - tftp-Daemon
- **umount*** - Programm zum Unmounten von Filesystemen
- **update*** - Daemon, der dafür sorgt, daß die Puffer in regelmäßigen Abständen auf die Festplatten geschrieben werden.
- **useradd*, userdel*, usermod*** - Programme zum Hinzufügen, Löschen und Ändern von Benutzern
- **utmp, wtmp** - Protokolldateien, in denen das Ein- und Ausloggen von Benutzern festgehalten wird.
- **xvmounttab** - Konfigurationsdatei für `xvmount`

13.2 Übersicht /etc-Verzeichnisse

- **default** - Verzeichnis, in dem Default-Parameter für den `useradd`-Befehl stehen
- **lilo** - Verzeichnis des Linux Loaders

- **skel** - Verzeichnis mit den Dateien, die bei useradd automatisch in das Home-Verzeichnis eines neuen Benutzers kopiert werden.

13.3 Konfiguration des Kernels

Die folgende Übersicht zeigt exemplarisch die Konfiguration des Linux Kernels. Da diese noch sehr stark im Wandel ist, sind Abweichungen möglich.

```
linux0:/tmp/linux> make config
/bin/sh Configure  < config.in
*
* General setup
*
Kernel math emulation (CONFIG_MATH_EMULATION) [y] n
Normal harddisk support (CONFIG_BLK_DEV_HD) [y]
XT harddisk support (CONFIG_BLK_DEV_XD) [n]
TCP/IP networking (CONFIG_INET) [y]
Limit memory to low 16MB (CONFIG_MAX_16M) [n]
System V IPC (CONFIG_SYSVIPC) [y]
Use -m486 flag for 486-specific optimizations (CONFIG_M486) [y]
*
* SCSI support
*
SCSI support? (CONFIG_SCSI) [n] y
*
* SCSI support type (disk, tape, CDrom)
*
Scsi disk support (CONFIG_BLK_DEV_SD) [n] y
Scsi tape support (CONFIG_BLK_DEV_ST) [n] y
Scsi CDROM support (CONFIG_BLK_DEV_SR)[n] y
*
* SCSI low-level drivers
*
Adaptec AHA1542 support (CONFIG_SCSI_AHA1542) [n] n
Adaptec AHA1740 support (CONFIG_SCSI_AHA1740) [n] y
Future Domain SCSI support (CONFIG_SCSI_FUTURE_DOMAIN) [n] n
Seagate ST-02 and Future Domain TMC-8xx SCSI support
(CONFIG_SCSI_SEAGATE) [n] n
UltraStor SCSI support (CONFIG_SCSI_ULTRASTOR) [n] n
7000FASST SCSI support (CONFIG_SCSI_7000FASST) [n] n
*
* Network device support
*
Network device support? (CONFIG_ETHERCARDS) [y]
SLIP (serial line) support (CONFIG_SLIP) [n] y
PLIP (parallel port) support (CONFIG_PLIP) [n] y
NE2000/NE1000 support (CONFIG_NE2000) [y] y
WD80*3 support (CONFIG_WD80x3) [y] n
3c503 support (CONFIG_EL2) [y] n
HP PCLAN support (CONFIG_HPLAN) [y] n
AT1500 and NE2100 support (CONFIG_AT1500) [y] n
D-Link DE600 pocket adaptor support (CONFIG_DE600) [y] n
*
Sony CDU31A CDROM driver support (CONFIG_CDU31A) [n] n
Mitsumi CDROM driver support (CONFIG_MCD) [n] n
```

```
*
* Filesystems
*
Standard (minix) fs support (CONFIG_MINIX_FS) [y]
Extended fs support (CONFIG_EXT_FS) [n] y
Second extended fs support (CONFIG_EXT2_FS) [n] y
xiafs filesystem support (CONFIG_XIA_FS) [n] y
msdos fs support (CONFIG_MSDOS_FS) [y] y
/proc filesystem support (CONFIG_PROC_FS) [y] y
NFS filesystem support (CONFIG_NFS_FS) [y] y
ISO9660 cdrom filesystem support (CONFIG_ISO9660_FS) [n] y
*
*  character devices
*
Keyboard meta-key sends ESC-prefix (CONFIG_KBD_META) [y] y
Keyboard Num Lock on by default (CONFIG_KBD_NUML) [y] y
Logitech busmouse support (CONFIG_BUSMOUSE) [n]
PS/2 mouse (aka "auxiliary device") support (CONFIG_PSMOUSE) [n]
Microsoft busmouse support (CONFIG_MS_BUSMOUSE) [n]
ATIXL busmouse support (CONFIG_ATIXL_BUSMOUSE) [n]
Selection (cut and paste for virtual consoles)
(CONFIG_SELECTION) [y]
QIC-02 tape support (CONFIG_TAPE_QIC02) [n]
*
* Sound
*
Sound card support (distributed separately) (CONFIG_SOUND) [n]
*
* Kernel hacking
*
Debug kmalloc/kfree (CONFIG_DEBUG_MALLOC) [n]
Kernel profiling support (CONFIG_PROFILE) [n]
Verbose scsi error reporting (kernel size +=12K)
(CONFIG_SCSI_CONSTANTS) [n]

The linux kernel is now hopefully configured for your setup.
Check the top-level Makefile for additional configuration,
and do a 'make dep ; make clean' if you want to be sure all
the files are correctly re-made

mv .config~ .config
make soundconf
cd kernel/chr_drv/sound;make config
```

13.4 FTP-Server

fgb1.fgb.mw.tu-muenchen.de	129.187.200.1	/pub/linux
ftp.informatik.tu-muenchen.de	131.159.0.110	/pub/Linux
ftp.dfv.rwth-aachen.de	137.226.4.105	/pub/linux
ftp.ibr.cs.tu-bs.de	134.169.34.15	/pub/os/linux
ftp.informatik.rwth-aachen.de	137.226.112.172	/pub/Linux
tsx-11.mit.edu	18.172.1.2	/pub/linux
sunsite.unc.edu	152.2.22.81	/pub/Linux
nic.funet.fi	128.214.6.100	/pub/OS/Linux
ftp.mcc.ac.uk	130.88.203.12	/pub/linux
src.doc.ic.ac.uk	146.169.2.1	/packages/linux
ftp.ibp.fr	132.227.60.2	/pub/linux
kirk.bond.edu.au	131.244.1.1	/pub/OS/Linux
ftp.uu.net	137.39.1.9	

wuarchive.wustl.edu	128.252.135.4	/mirrors/linux
ftp.win.tue.nl	131.155.70.100	/pub/linux
ftp.stack.urc.tue.nl	131.155.2.71	/pub/linux
srawgw.sra.co.jp		/Linux
cair.kaist.ac.kr		/pub/Linux
ftp.denet.dk	129.142.6.74	/pub/OS/linux

13.5 Mailboxen

FORMEL-Box 04191.2846 ZyX 16.8k 5:30-03:30
Linux Server/A. Braukmann 441.592-963 ZyX 16.8k
MM's Spielebox 05323.3515 ZyX 16.8k V32b
MM's Spielebox 05323.3540 9.6k
UB-HOFF /A. Hoffmann 0203.584-155 ZyX+ 19.2k
Zaphods BBS/C.Lueders
0228.262-894 HST V32b
0228.229-161 ZyX+ 19.2k V32b
0228.229-147 2400/MNP4
ISDN 0228.911-10-41 64.0k BPS V110/X75
CS-Port /C. Schmidt 030.491-34-18 ZyX+ 19.2k V32b
BigBrother /R. Gmelch 030.335-63-28 ZyX 16.8k V32b
16:00-23:00 MEZ
CRYSTAL BBS 07152.240-86 HST 14.4k
Echoblaster BBS #2 07142.212-35 V32b 8-20:00,23-2:00

13.6 Weitere Literatur

Andeleigh, Prabhat K.
UNIX System Architecture. Prentice Hall [1993]

Comer, Douglas E.
Internetworking with TCP/IP. Vol. 1-3, Prentice Hall [1991]

Flanagan, David
X Toolkit Intrinsics Reference Manual. Vol. 5, O'Reilly [1993]

Fuchssteiner B.
MuPAD - Benutzerhandbuch. Birkhäuser [1993]

Gilly, Daniel
UNIX in a Nutshell. O'Reilly [1992]

Gulbins, Jürgen
UNIX. Springer [1988]

Heller, Dan
XView Programming Manual. Vol. 7, O'Reilly [1991]

Heller, Dan
Motif Programming Manual. Vol. 6, O'Reilly [1992]

Hewlett Packard (Hrsg.)
Ultimate Guide to the Vi and Ex Text Editors.
Addison Wesley [1990]

Kernighan, Pike
Der UNIX-Werkzeugkasten. Hanser [1986]

Kopka, Helmut
LaTeX - Eine Einführung. Addison Wesley [1992]

Lippmann, Stanley B.
C++ Primer. Addison Wesley [1991]

Mui, Linda und Pearce, Eric
X Window System Administrator's Guide. Vol. 8, O'Reilly [1993]

Nye, Adrian
Xlib Programming Manual. Vol. 1, O'Reilly [1992]

Nye, Adrian
Xlib Reference Manual. Vol. 2, O'Reilly [1992]

Nye, Adrian und O'Reilly, Tim
X Toolkit Intrinsics Programming Manual.
Vol. 4, O'Reilly [1990]

Open Software Foundation
OSF/Motif Progammer's Guide. Prentice Hall [1993]

Open Software Foundation
OSF/Motif Progammer's Reference. Prentice Hall [1993]

Open Software Foundation
OSF/Motif Users's Guide. Prentice Hall [1993]

Quercia, Valerie und O'Reilly, Tim
X Window System User's Guide. Vol. 3, O'Reilly [1991]

Sanitfaller M.
TCP/IP und NFS in Theorie und Praxis. Addison Wesley [1993]

Schoonover, Michael A.
GNU Emacs - UNIX Text Editing and Programming.
Addison Wesley [1992]

Sebastian, Hetze und Müller, Martin
Linux Anwender Handbuch. Selbstverlag [1993]

Stapelberg, Stefan
UNIX System VR4 für Einsteiger und Fortgeschrittene.
Addison Wesley [1993]

Stevens, W. Richard
Advanced Programming in the UNIX Environment.
Addison Wesley [1992]

Stevens, W. Richard
UNIX Network Programming. Prentice Hall [1990]

Young, Douglas A.
X Window System, Programming and Applications with Xt .
Prentice Hall [1990]

Index

Springer-Verlag und Umwelt

Als internationaler wissenschaftlicher Verlag sind wir uns unserer besonderen Verpflichtung der Umwelt gegenüber bewußt und beziehen umweltorientierte Grundsätze in Unternehmensentscheidungen mit ein.

Von unseren Geschäftspartnern (Druckereien, Papierfabriken, Verpakkungsherstellern usw.) verlangen wir, daß sie sowohl beim Herstellungsprozeß selbst als auch beim Einsatz der zur Verwendung kommenden Materialien ökologische Gesichtspunkte berücksichtigen.

Das für dieses Buch verwendete Papier ist aus chlorfrei bzw. chlorarm hergestelltem Zellstoff gefertigt und im pH-Wert neutral.